MOTION TRACKING PROBLEMS IN INTERNET OF THINGS (IOT) AND WIRELESS NETWORKING

BY

MAHANTH KRISHNAPPA GOWDA

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Doctoral Committee:

Professor Romit Roy Choudhury, Chair
Professor Nitin Vaidya
Professor Bruce Maggs, Duke University
Professor Klara Nahrstedt
Professor Tarek Abdelzaher

Abstract

The dissertation focuses on inferring various motion patterns of internet-of-things (IoT) devices, by leveraging inertial sensors embedded in these objects, as well as wireless signals emitted (or reflected) from them. For instance, we use a combination of GPS signals and inertial sensors on drones to precisely track its 3D orientation over time, ultimately improving safety against failures and crashes. In another application in sports analytics, we embed sensors and radios inside baseballs and cricket balls and compute their 3D trajectory and spin patterns, even when they move at extremely high speeds. In a third application for wireless networks, we explore the possibility of physically moving wireless infrastructure like Access Points and base-stations on robots and drones for enhancing the network performance. While these are diverse applications in drones, sports analytics, and wireless networks, the common theme underlying the research is in the development of the core motion-related building blocks. Specifically, we emphasize the philosophy of "fusion of multi modal sensor data with application specific model" as the design principle for building the next generation of diverse IoT applications. To this end, we draw on theoretical techniques in wireless communication, signal processing, and statistics, but translate them to completely functional systems on real-world platforms.

To Gowramma and Krishnappa.

Acknowledgments

I am deeply indebted to my advisor Prof. Romit Roy Choudhury for being a constant source of motivation, encouragement and guidance throughout my Ph.D. journey. The thesis is a result of countless hours of his support and mentoring across all dimensions. He provided me an opportunity and freedom to pursue multiple exciting projects. His advise is not only limited to research, but also towards holistic development of a student. I couldn't have wished for anything more from him. He has inspired me to embark on a career in Academia.

I am thankful to my committee members Prof. Bruce Maggs, Prof. Nitin Vaidya, Prof. Klara Nahrstedt, and Prof. Tarek Abdelzaher for their insightful comments, constructive criticisms and invaluable feedback at various stages of my Ph.D. Their support has been immensely helpful not only for shaping the thesis but also for my academic job search. I am also grateful to Sung-Ju Lee at KAIST for a productive collaboration on "Enterprise Wireless Networks", as well as for his invaluable feedback about my work.

The work in this thesis is a result of active collaboration with several co-authors. I had a thoroughly enjoyable experience during my internship with Xue Yang at the wearable computing group at Intel. I would like to express gratitude for her technical support as well as providing access to resources for our project on sports analytics. The work on "drone orientation tracking" is a result of collaboration with Justin Manweiler at IBM research. Thanks for an energetic and fun-filled internship experience at IBM as well as the work itself. Immensely thankful to lab-mates and friends Ashutosh Dhekne and Sheng Shen. I have co-authored five papers with Ashutosh. From intense brainstorming discussions to flying drones in snowy Illinois weather,

the thesis wouldn't be the same without the strong teamwork with Ashutosh. Sheng collaborated with me on cricket ball tracking and human arm tracking projects. It was a pleasure to work with Sheng and the discussions with him have always been intellectually stimulating. I am thankful to Srihari Nelakuditi from USC not only for collaborating with me on wireless networking projects but also for making himself available for technical and non-technical conversations. I also acknowledge the contributions of other co-authors Bozidar Radunovic, Souvik Sen, Xuan Bao, Songchun Fan, Nirupam Roy with whom I have worked on several exciting projects which may not be a part of the thesis.

I acknowledge the support of my parents, Gowramma and Krishnappa, and brother Hemanth Gowda during the long journey, for backing me in every career decision. They are more excited than me about my research. I would also like to thank my teachers at Jindal Public School - Bangalore for inspiring me during school days, as well as faculty at IIT-BHU and Duke for providing me a solid foundation and mentoring during my early career. Last but not the least, sincere regards to all current and past lab-mates and friends for making the journey a pleasantly memorable experience.

Table of Contents

Chapter 1

Introduction

Sensing, computing and communication are rapidly percolating into human lives, blurring the line between real and digital worlds [1–5]. In addition to progress in hardware, algorithmic advances in machine learning, robotics, and signal processing has enabled internet of things (IoT) devices to be miniaturized and power efficient to an extent that we can embed them into everyday objects and inject intelligence. For instance, we have several innovative applications in smart and connected health-care, personal robotics, self-driving cars, drones, smart homes, precision agriculture etc. In the above applications, note that the IoT devices are not only passive sensing devices but also actively interact and manipulate the environment (Actuation). Actuation in the form of physical mobility is an integral part of most of these applications, thus making mobility tracking and localization important problems.

The dissertation develops mobility inferencing techniques from wireless signals and mobile sensors for a variety of emerging IoT applications. We specifically focus on inferring motion patterns of objects and humans, by leveraging inertial sensors embedded in these objects, as well as wireless signals emitted (or reflected) from them. For instance, we combine GPS signals and inertial sensors on drones (Figure 1.1(a)) to precisely track its 3D orientation over time, ultimately improving safety against failures and crashes. In another application in sports analytics, we embed sensors and radios inside baseballs and cricket balls (Figure 1.1(b) and (c)), and computed their 3D trajectory and spin patterns, even when they move at extremely high speeds. In a third application for wireless networks, we explore the possibility of physically moving wireless infrastructure like Access Points and base-stations on robots and drones for enhancing the network performance. While these are diverse applications in drones, sports

analytics, and wireless networks , the common theme underlying the research is in developing the core motion-related building blocks.

Extensive literature exists in the areas of motion tracking and localization from various disciplines including robotics, control theory, signal processing and mobile computing [6–17]. However, the nature of emerging IoT applications impose completely unpredictable challenges and open up novel problems never imagined before, thereby entailing a fresh perspective. Fortunately, the applications also provide unique opportunities for catering to their specific requirements. Through a series of examples, the dissertation emphasizes the philosophy of "fusion of multi-modal sensory data with application specific models" towards solving mobility inferencing problems. To this end, we draw on theoretical techniques in wireless communication, signal processing, and statistics, but translate them to completely functional systems on real-world platforms. We briefly elaborate on the IoT applications presented in this dissertation and their technical contributions below.

Figure 1.1: (a) Drone mounted with redundant GPS modules (b) Cricket ball embedded with sensors (c) UWB MIMO Anchors for ball tracking

1.1 Mobility tracking applications

Drone Orientation Tracking

Inertial sensors continuously track the 3D orientation of a flying drone, serving as the bedrock for maneuvers and stabilization. However, even the best inertial measurement units (IMU) are prone to various types of correlated failures. In chapter 2, we present *SafetyNet*, a system that augments drones with multiple GPS receivers to provide a failsafe mechanism for IMU failures (Figure 1.1(a)). The core challenge is in accurately computing the relative locations between GPS receiver pairs, and translating these measurements into the drone's 3D orientation. This is a non-trivial problem given basic GPS is only 3 meter accurate, several times higher than the size of the drone itself. While GPS carrier phases provide high precision, sub-centimeter ranging information, they are corrupted by integer errors (ambiguity) in multiples of GPS wavelength. Moreover, GPS-based orientation needs to be precise even under sharp drone maneuvers, GPS signal blockage, and sudden bouts of missing data. *SafetyNet* proposes a novel inferencing technique by fusing carrier phase differences over time and space into a probabilistic filtering problem. While the computational expenses of filtering can be large, we exploit the geometric model of GPS receiver placement on a small drone. This enables *SafetyNet* to narrow down the search space in the presence of ambiguities and enable real time tracking. Extensive flight tests demonstrate an accuracy comparable to IMU even under cloudy, windy and foggy weather conditions.

Sports Analytics

In chapter 3, we propose a system called *iBall* that explores the possibility of bringing IoT to sports analytics, particularly to the game of Cricket. *iBall* develops solutions to track a ball's 3D trajectory and spin with inexpensive sensors and radios embedded in the ball (Figure 1.1(b) and (c)). Unique challenges arise due to high speed ball motion, sparse availability of sensor

data, and free-fall pattern of ball motion. This renders existing localization and motion tracking solutions inadequate. Fortunately, the ball motion obeys well defined principles of physics, and *iBall* exploits those motion models. Specifically, *iBall* fuses disparate sources of partial information – wireless, inertial sensing, and physics motion models – into a non-linear error minimization framework. Measured against Vicon ground truth, *iBall* achieves cm level tracking accuracy. The results do not rely on any calibration or training, hence the core techniques can extend to other sports like baseball, with some domain-specific modifications. Patterns of balls, racquets, and players are being analyzed for coaching, strategic insights, and predictions. *iBall* provides a significantly cheaper alternative to million dollar camera based solutions. Real-time analytics should be possible anytime, anywhere. Aspiring players in local clubs could read out their own performance from their smartphone screens; school coaches could offer quantifiable feedback to their students.

Mobile Base-stations

In chapters 4 and 5, we explore the possibility of injecting mobility into the wireless infrastructure. *What if network infrastructure of the future – WiFi Access Point (AP)s, enterprise Wireless LANS, cell towers – are empowered with the ability to move physically?* Mobility is expected to bring a new degree of freedom (DoF) to network design, but more importantly, this DoF compliments existing dimensions of wireless innovation. Techniques for power control, channel

Figure 1.2: (a) Roomba Robot with Laptop and Software Radio (b) Drone carrying a WiFi AP

allocation, MIMO, localization, topology control, can all benefit if APs are able to move, even

in the scale of inches. Advances in personal robotics, beginning from the popular Roomba to the more recent quadcopters are already mainstream. Robotic technology does not seem like a fundamental barrier. In chapter 4, we present *iMob*, that makes a case for *robotic wireless networks*. By using Roomba robots with software radios and laptops (Figure 1.2(a)), *iMob* shows that even cm scale mobility (while being tethered to Ethernet and power) can offer substantial gains. In an outdoor setting, chapter 5 presents *DroneNet*, a system that creates flying base-stations using quadcopters (Figure 1.2(b)) to serve as an extender to cell towers. While the challenge in both cases is to quickly find the optimal position for the mobile AP, *iMob* and *DroneNet* combine sensed wireless data with wireless channel models. Specifically, *iMob* exploits statistical variation principles of wireless indoor multipath, whereas *DroneNet* combines measured channel strengths with environment aware 3D propagation models to considerably reduce the search space for positioning.

1.2 Core technical contributions

Adjusted particle filter

In *Safetynet*, the core challenge is to obtain precise location measurements using carrier phase measurements of Radio Frequency (RF) signals. Consider N wireless receivers, arranged in a rigid geometry, and K wireless transmitters sending RF signals at a certain frequency. *SafetyNet* considers a specific instantiation of the above case where the wireless receivers are GPS receivers and transmitters are GPS satellites for an application in drones. Multiple GPS receivers are placed on different arms of the drone as shown in Figure 1.3. By tracking the relative position vectors $\vec{v}_i$ between the wireless receivers, we can determine the orientation of the rigid body (drone in *SafetyNet*). The key challenge here is to accurately track $\vec{v}_i$.

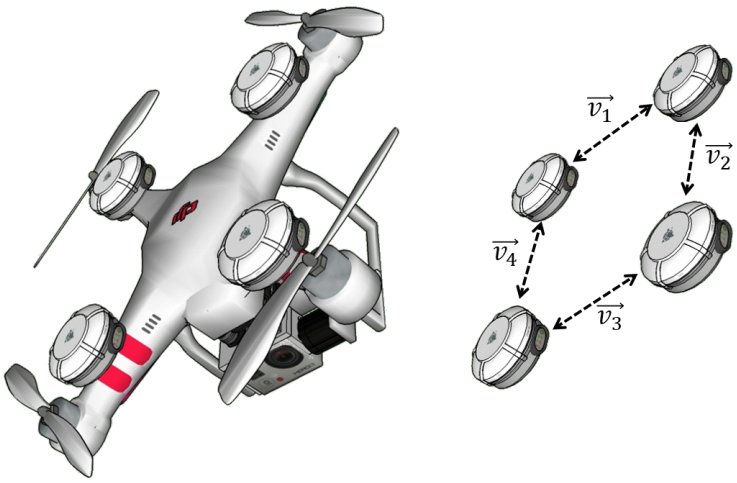

Figure 1.3: Drone orientation can be determined by tracking the time varying relative position vectors $\vec{v}_i$ between GPS receivers

Towards the goal of computing the relative vectors, we exploit carrier phase measurements, which provide high precision location information. As depicted in Figure. 1.4, the true range between a receiver, and a transmitter can be expressed in terms of sinusoidal signals at the transmission frequency. The true range can be divided into an integer number of cycles (integer ambiguity) and a fractional part.

$$True\ Range = \lambda N + Carrier\ Phase \tag{1.1}$$

$$Carrier\ Phase = \lambda N - True\ Range \tag{1.2}$$

The fractional part is what the receivers can measure, called as *Carrier Phases*. These phases together with the knowledge of the integer ambiguity can provide high precision ranging information leading to accurate orientation estimates. However, tracking integer ambiguity is the key challenge in leveraging the *Carrier Phases*, and our techniques are focused on estimating the integer ambiguity.

In reality, tracking integer ambiguity for absolute positioning is impossible due to clock syn-

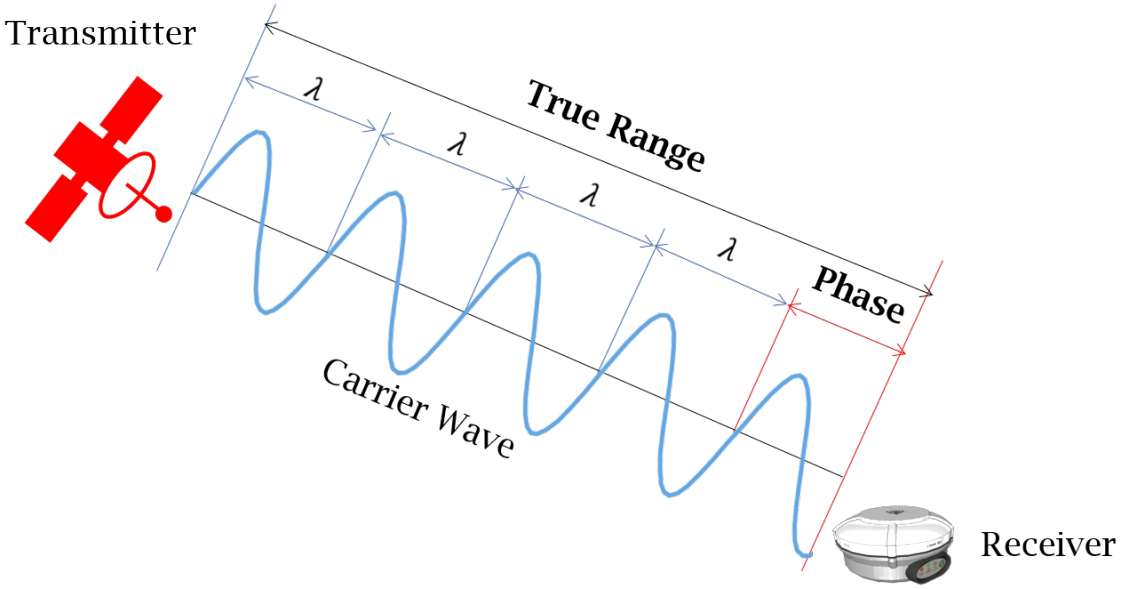

Figure 1.4: Carrier phases

chronization errors between transmitters and receivers. However, since we are only interested in relative positioning, we can eliminate most of the synchronization errors by performing differencing operations between phases across receivers (More details in chapter 2). Briefly, there are two kinds of differentials.

- *Spatial Differentials*: Differencing carrier phases over multiple receivers can provide information about the orientation. However, it is polluted by integer ambiguities.
- *Temporal Differentials*: Differencing of carrier phases over time, can give information about the change in orientation.Under low dynamic motion, these observables are not polluted by integer ambiguities, however, under high dynamic conditions, loss of synchronization with transmitters, can introduce integer ambiguity related errors called cyclic clips.

We use a Bayesian filtering approach to combine the information from the above two perspec-

tives, to estimate the most likely orientation. As shown in Figure.1.5, we model the state of the filter as the orientation. The temporal and spatial differentials form the transition and measurement functions of the filter, thus capturing orientation change information, and absolute orientation information respectively.

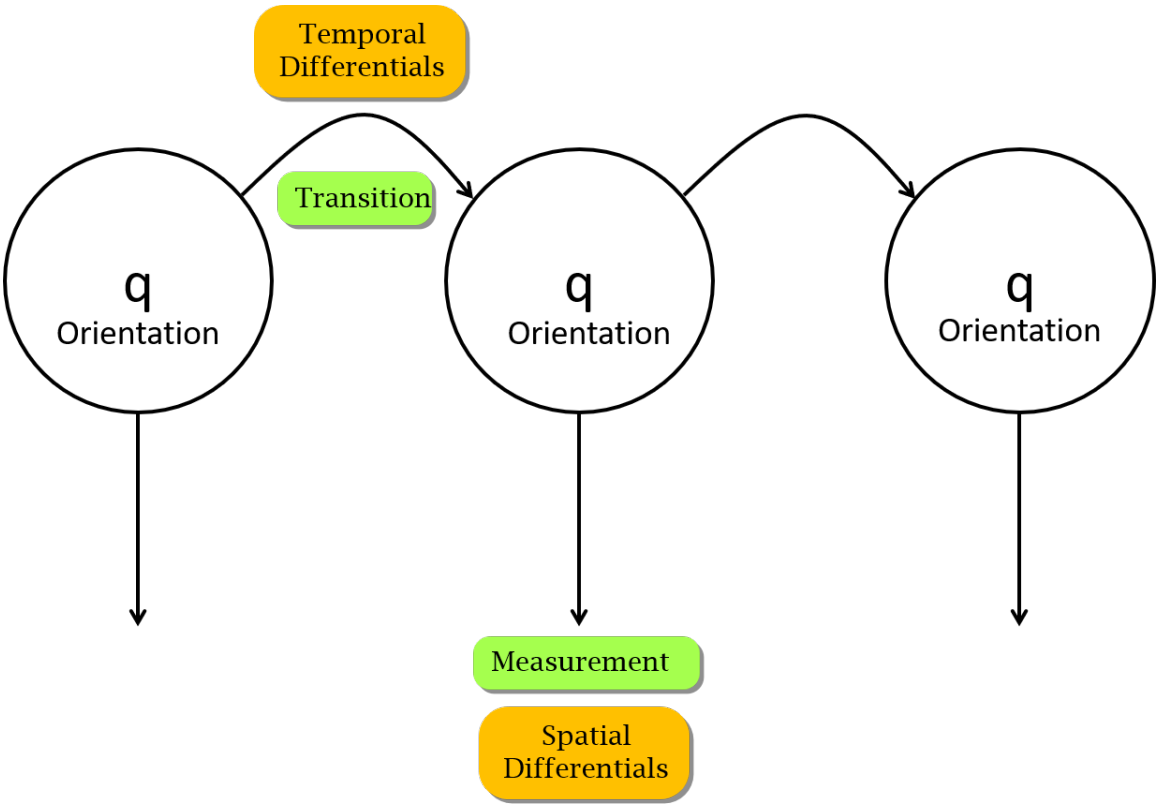

Figure 1.5: Bayesian filter for orientation tracking

The main source of error comes from integer ambiguities - a multi-modal error distribution. Thus, we use particle filters to track states corresponding to various possible integer ambiguities.

A conventional particle filter converges slowly and needs high computational resources which is infeasible for real time operation. Thus, we propose an improvement to the particle filter design. The opportunity arises from the observation that in the conventional particle filter, a par-

ticle's weight is typically proportional to its likelihood function (i.e., proximity to the true state). Thus, when the system resamples, the concentration of particles are proportionally greater at the higher weight particles. Over time, the hope is that one of the particles would converge to the correct state. However, we ask: *why not move the particle to a state that maximizes the likelihood*. In fact, given that the error variances of the transition and measurement functions are known, the best estimate can be computed as a combination of the two. We could move each particle to the "best" state in its neighborhood, and then perform the resampling step.

Figure 1.6 illustrates the idea for a case of three particles. The middle particle is propagated to time $time_1$ using the transition function. Similarly, the measurement function produces an estimate of the new state. The transitioned and measured states along with their error distributions are depicted. The two estimates are combined using a Kalman filter like approach, and the particle "adjusted" to this best state. This particle is then propagated further to time $time_2$, which now has a smaller error variance.

We note that each particle essentially tracks a particular set of integer ambiguity vectors. As a result, every carrier-phase measurement yields distinct orientation for each of these vectors. The benefits arise because: (1) For the correct integer ambiguity vector, the error properties of the measurement function becomes Gaussian. Hence, the combining process indeed is like a Kalman filter, leading to faster convergence. (2) For incorrect integer ambiguity vectors, the gaussian error properties will not hold. Moreover, the transitioned particle will be adjusted towards a wrong state because of incorrect integer ambiguity resolution. Thus, the net outcome is faster convergence for the correct ambiguities, while disappearing particles for incorrect ambiguities. More details and efficacy of the design is presented in Chapter 2.

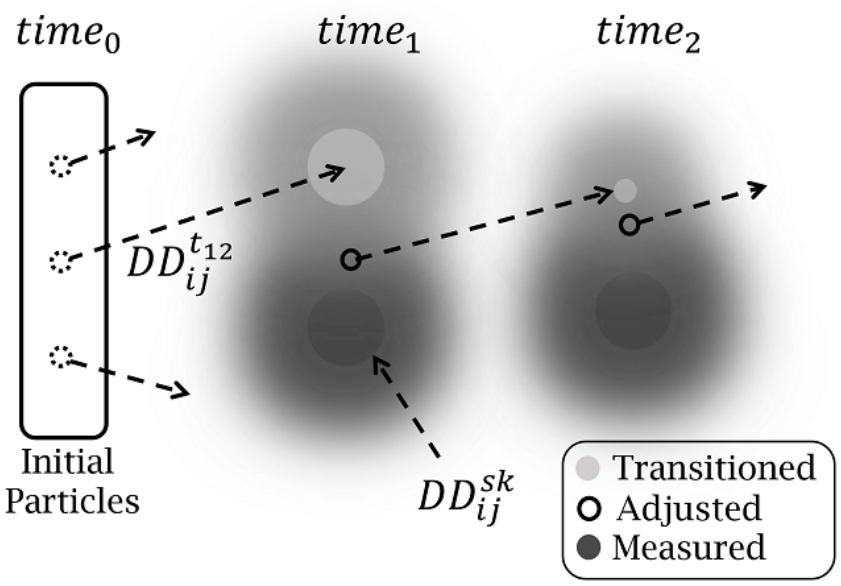

Figure 1.6: Adjusted Particle Filter (APF) results in faster reliable convergence even with few particles.

Free fall motion tracking

The core contribution in *iBall*, our work in sports analytics lies in tracking the location and orientation of freely falling objects, particularly under high speed translational and rotational motion. Conventional techniques for tracking motion of smartphones, robots, drones etc completely fail under these conditions. Our core opportunity comes from the fact that free fall motion follows well defined laws of physics, thus providing convenient models for tracking motion.

Rotation tracking: We use inertial sensors consisting of accelerometers, gyroscopes, and magnetometers for tracking because of their wide applicability. Accelerometers provide important orientation related information on smartphones. The spring of the accelerometer reacts to gravity, and the direction of the tension in the spring provides orientation information. How-

ever, the accelerometer in a freely falling object does not experience the reactive force(Figure 1.7), and hence cannot measure orientation. A gyroscope on the other hand saturates at 5 rev-

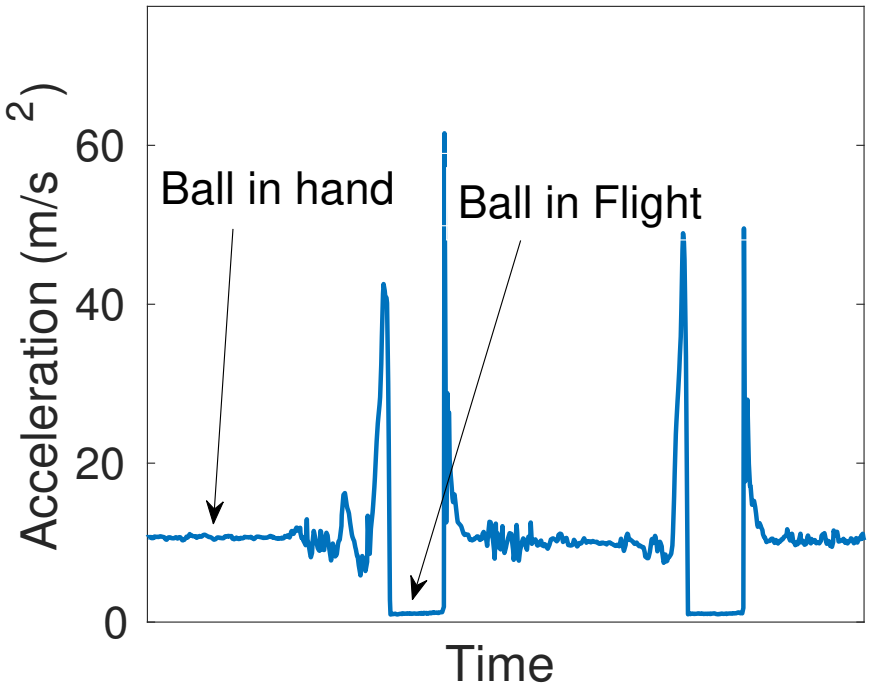

Figure 1.7: Accelerometer measurement drops to zero under free fall

olutions per second and cannot measure high speed rotations (Figure 1.8). Hence, we are only left with magnetometers for tracking the orientation. The core opportunity that helps us work with under constrained measurement obtained from magnetometer is the model of rotational motion. Consider figure 1.9. For the specific case of a spinning cricket ball, it shows the trajectory (green line) of the axis of rotation (red line) over the duration of a ball flight. Given slow change in the axis of rotation, the tip of the red line can be modeled using a two dimensional state – azimuth θ_t, and elevation ϕ_t – which varies slowly with time.

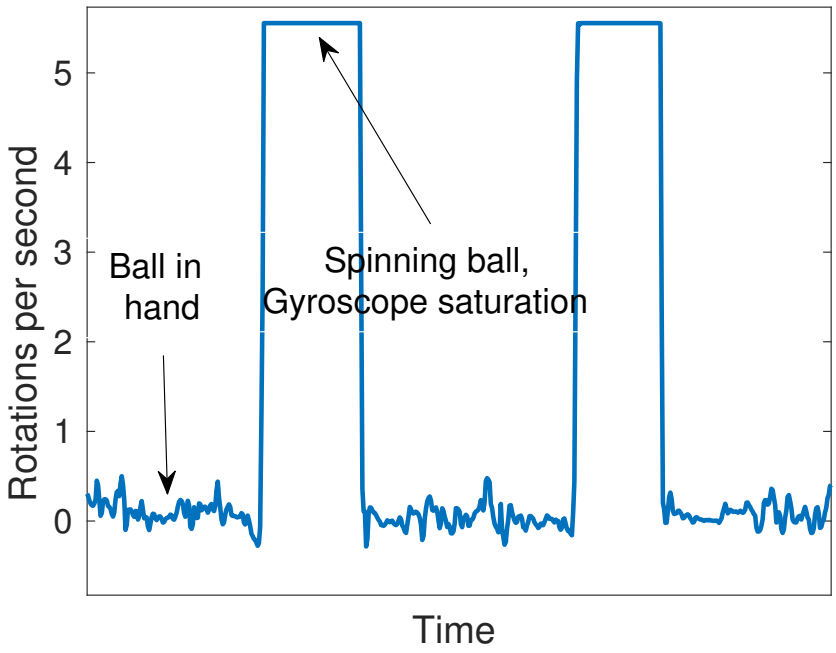

Figure 1.8: Gyroscope measurement saturates very quickly under high spin

$$\theta_t = A_{el}\,t^2 + B_{el}\,t + C_{el} \tag{1.3}$$

$$\phi_t = A_{az}\,t^2 + B_{az}\,t + C_{az} \tag{1.4}$$

Magnetometer measurements (North direction) traces circles around the red line. By performing a constrained optimization with the above constraint, we find the best set of model parameters (A_{el}, A_{az}, B_{el}, B_{az}, C_{el}, C_{az}) to find θ_t, ϕ_t thus giving us the instantaneous axis of rotation as well as orientation (more details in Chapter 3).

Location tracking: Consider Figure 1.10. We use two ultra-wideband (UWB) radios to perform time of flight measurements (blue lines) with a UWB radio embedded inside a freely falling object. Practical constraints may allow under-constrained, sparse measurements. For example,

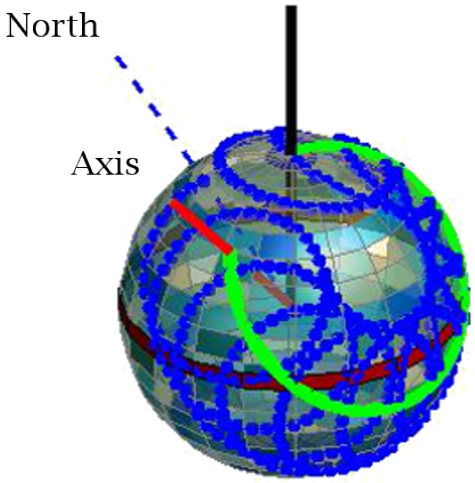

Figure 1.9: Rotation axis changes slowly during the flight of the ball

only two UWB radios are allowed in cricket, placed conveniently behind stumps as shown in Figure 1.10. Also, the sampling rate of ranging measurements is very less in comparison with the speed of ball motion). The bottom line is that the UWB radios can only localize the ball to lie within a circle, and cannot determine the precise 3D location. Figure 1.10 shows circles of possible ball locations obtained at various points on the ball trajectory. The core opportunity that helps us estimate the 3D location even with under-constrained, sparse UWB measurements is the parabolic nature of the ball trajectory. Since the external forces acting on the ball under free fall are minimal, trajectory (red line) can be modeled as a parabola as shown below.

$$S_{xe}(t) = x_o + v_x t \tag{1.5}$$

$$S_{ye}(t) = y_o + v_y t \tag{1.6}$$

$$S_{ze}(t) = z_o + v_z t - 0.5 g t^2 \tag{1.7}$$

S_{xe} is the x-coordinate of the ball at time t, x_o is the initial x-coordinate, and v_x is the initial velocity. The other variables are analogous in y and z co-ordinates, g is acceleration due to gravity.

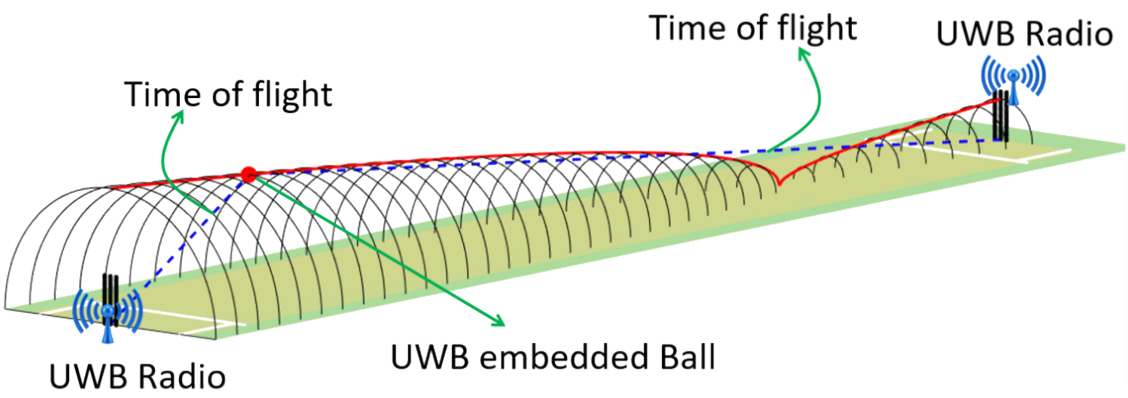

Figure 1.10: Location tracking with under-constrained time of flight measurements from UWB radios

By modeling the above constraints into an optimization problem, we can track the 3D location of the ball with promising accuracy even with under constrained, sparse UWB measurements. (More details in chapter 3).

Wireless channel modeling and prediction

Indoors: . Our robotic AP system in *iMob* shows that even feet scale micro mobility of the AP can offer a substantial throughput gain. The optimal location for an AP to park itself depends upon the channel properties of its clients, neighboring APs and interferers. We design a metric which is a function of the above, and move the AP to park itself in a location that maximizes the metric. Given that a WiFi channel can change in the granularity of $3cm$, the search process can be slow. Practical issues like inability to navigate a robot to a precisely optimal $3cm$ wide location, inability to stop quickly while detecting a high metric location introduces challenges. To solve these challenges, *iMob* borrows from optimal stopping theory [18, 19] in statistics. *iMob* explores the first few positions quickly to get a rough spatial distribution of the optimization metric. It would then move slowly and park itself in a position where the detected metric is above the 90%-ile of the distribution.

Outdoors: The core goal in *DroneNet* is to hover the drone at a location that maximizes client

throughputs. This is an expensive search problem because the best location is a function of complex environment including buildings, trees, and client locations. A brute force solution would be to fly the drone and conduct SNR measurements to empirically search for the optimal location. However, a large 3D search space – say 2 or 3 city blocks in Chicago – makes this approach prohibitively time consuming. Movements in clients and changes in traffic patterns will occur at faster time scales, rendering this brute force search useless. Hence we require a solution that is lightweight and quick. Simple strategies like hovering at the centroid of a group of clients are unsuitable due to the non-monotonous relation between distance and SNR. Results from historical searches are also not useful since each new situation is somewhat unique in its placement of clients and the type of traffic demands.

We explore the possibility of using RF ray tracing as a hint to narrow down the scope of search. Our key idea is to model the dominant structures located in an area—such as the buildings and trees—to roughly model the terrain of the region, and then simulate how signals would bounce and scatter from the drone to the various clients. Of course, such simulations yield coarse-grained results since the simulated SNR is sensitive to centimeter-scale errors. Nonetheless, we find that these simulated results can still be valuable in broadly guiding the drone towards the right direction, i.e., towards areas where the SNR is relatively better. Once the drone arrives in this area, it physically conducts measurements to fine-tune its hovering location. Given a considerably smaller search space, the operation incurs far less time. The drone now hovers at this location offering connectivity to clients. If client positions or traffic changes substantially, the drone recomputes the ray-tracing results and finds a new hovering location.

The rest of the dissertation expands on each of the applications in motion tracking and associated techniques in detail. Finally, we conclude by summarizing the contributions as well as discuss possible future extensions.

Chapter 2

Tracking Drone Orientation with Multiple GPS Receivers

2.1 Introduction

Despite excitement, we are not yet ready for a world of drones flying among people. Challenges of liability, regulation, and limited societal acceptance all arise because drones are still too dangerous for responsible use in public places. A capstone obstacle is the untrustworthiness of inertial sensors [20–24].

A drone's inertial sensors, also called the Inertial Measurement Unit (IMU), is akin to a human's inner ear. Without a good awareness of orientation and balance, a drone cannot effectively modulate current to its rotors – even hovering becomes impossible. In the best case, the drone may not take off, or fall vertically from the sky. However, the worse case is when the drone tries to correct for its poor sensory inputs, and accelerates uncontrollably in an unintended direction with even greater force. For a package delivery drone measured in 10s of kilograms, this might be deadly.

A natural response is to install redundant IMUs (i.e., accelerometer, gyroscope, and compass). Given that each sensor is relatively small, light, and inexpensive, this is an easy fix – some commercial drones have already adopted this [25, 26]. Unfortunately, redundancy only addresses unreliable hardware. Correlated noise sources, including motor vibration, electromagnetic interference, and ambient ferromagnetic influences, remain as serious concerns [27–31].

To provide stronger IMU failover guarantees, we consider a GPS based approach *completely*

orthogonal to the fundamental nature of IMU sensing. We install 4 GPS receivers at the corners of a single drone – upon IMU failure, we utilize these GPSs to estimate the drone's 3D orientation. Put differently, *we apply GPS to estimate pitch, roll, and yaw, without any inertial or magnetometer assistance.*

The drone's pitch, roll, and yaw can be expressed as a function of the pairwise 3D vectors between the GPS receivers – as the drone flies, these vectors change with respect to the Earth's reference frame. SafetyNet's core task is to precisely estimate these 3D vectors. To achieve IMU-like accuracy ($\approx 1^{\circ}$), the vectors need to be estimated at centimeter scale precision. This precision needs to be upheld constantly over time, across agile flight patterns, poor satellite SNR, and even short periods of missing data.

Recent research [32, 33] has improved on Differential GPS (DGPS) to achieve $\approx 15cm$ error for relative localization. This is adequate for many on-road vehicular applications, such as lane change detection, peer-to-peer car coordination, etc. However, the accuracy requirement for drone orientation is much higher. Multi-GPS orientation estimation techniques have been proposed for ships, planes and low dynamic drone flights [9, 10, 34]. However, extending them to commercial drones of small form factor and rapidly changing orientation is non-trivial and challenging. The demands are far stricter, both in terms of accuracy and time-granularity of tracking. Finally, the estimation techniques must be lightweight to be able to serve as a real-time IMU replacement during flights.

SafetyNet appropriately adopts ideas and techniques from the mature GPS literature, and builds over them to mitigate the challenges. The core additions, although closely interspersed with existing techniques, can be distilled as follows:

(1) Manipulating measurements across pairs of GPS receivers, satellites, and consecutive time

points, to ultimately capture the 3D orientation of the drone from 2 different perspectives. Using these 2 perspectives to formulate an estimation problem, amenable to Kalman Filtering (KF). While GPS literature is fraught with KF and measurement manipulations (called "double differentials"), SafetyNet combines the double differentials in a way that is novel to the best of our knowledge.

(2) GPS phase measurements are composed of an unknown portion, called "integer ambiguity". In attempting to resolve this ambiguity, past work have adopted techniques akin to "hard decoding", where the most likely state-estimate is propagated across time. We design the equivalent of "soft decoding", whereby top-K possibilities of the ambiguity are propagated, each associated with an inferred probability. A particle filter is used to execute this idea – the particle filter degenerating back into the Kalman Filter when the ambiguity is resolved confidently.

(3) We break away from the classical particle filter approach and "adjust" the state of the particles based on available measurements. This speeds up convergence of the system, while requiring fewer particles (considerably reducing the computational complexity). As a result, the overall SafetyNet system lends itself to real time operation on today's drone hardware.

(4) Finally, SafetyNet is a complete system borne out of significant engineering effort, including carrier-phase outlier detection, integration of multi-satellite systems for robustness, automatic baseline calibration, etc.

Our evaluation platform is composed of a 3DR octocopter (8 rotor) mounted with 4 off-the-shelf NEO-M8T GPS receivers, all connected to a Raspberry Pi via USB. A GoPro camera is mounted in the underbelly of the drone, facing vertically downward – vision data serves as the ground-truth for both IMU and GPS. We present extensive experimental data from 11 aggres-

sive flight sessions, performed under a wide range of weather conditions. Critically, our results demonstrate comparable accuracy to IMU even under the most aggressive aerial maneuvers within the capability of our drone.

Some Natural Questions

Is power consumption excessive with 4 GPS receivers? While GPS receivers are indeed considered power hungry on smartphones [35, 36], we note that the drone's physical flight requires 1000x more power, requiring a separate LiPO battery [37] weighing 804 grams. Hence the marginal increase in power from GPS is negligible.

Do GPS receivers add to the cost of drones? SafetyNet is mostly needed for bigger drones carrying heavier payloads (costly cameras, delivery packages). The cost of such drones and their payloads easily justify the additional GPS cost. Moreover, commercial drones already provide multiple GPSes for redundancy [38] – we believe the additional GPS cost should not be an issue.

What is the on-board IMU's orientation accuracy? How does it compare to SafetyNet? Results in this chapter will demonstrate comparable accuracy distributions between SafetyNet and the on-board IMU, even at the 99^{th} percentile. To be specific, pitch and roll are slightly worse with SafetyNet, but yaw is slightly better. This level of accuracy is adequate as a fallback mechanism – upon IMU failure, the drone could avoid aggressive maneuvers and fly back safely to its base.

What if GPS also fails? Crash reports typically indicate IMU failures [23]. This is because engine vibrations cause drift in IMU sensors, accumulating error over time [27,30,31]. Drone electronics might interfere with magnetometers [28, 29, 39]. However, none of these error sources affect GPS. While GPS could still fail due to deliberate jamming or extreme weather conditions, the possibility of both failing is naturally lower.

The rest of the chapter expands on these techniques, experimental findings, and contributions. We begin with an abridged technical primer on GPS processing (mostly derived from Differential GPS and related techniques), followed by system design, error mitigation, and an end-to-end system evaluation.

2.2 GPS Foundations

We present GPS foundations from first principles and end with discussions on modern techniques. As a result, this section is long. However, given that this chapter builds over core GPS algorithms, the material is necessary. We also believe the material is easy to follow.

2.2.1 Global Navigation Satellite System (GNSS)

GNSS is the generic name given to satellite systems that provide localization services to receiver's on earth. The Global Positioning System (GPS) [40] is one example of a GNSS, developed by the US Government during 1970-80s. GPS consists of a constellation of 31 satellites orbiting the earth at a height of 20,000 km. The satellites are simultaneously and continuously transmitting unique pseudo-random noise (PRN) sequences using CDMA at 2 different frequencies – 1575.42 (L1) and 1227.60 MHz (L2). They also broadcast ephemeris data using which the (satellite) position and time of transmission can be calculated. A GPS receiver on the ground localizes itself by trilateration, i.e., measuring and combining the time-of-flight (ToF) of PRNs from different satellites. Velocity is computed from the doppler shifts from each of the satellites.

GLObal NAvigation Satellite System (GLONASS) is another GNSS system launched by Russia in the 1980's [41]. Similarly, GALILEO [42] is an European GNSS system currently under development. Many GNSS receivers are capable of decoding signals from multiple GNSS systems,

providing increased accuracy. This project will also use such receivers and exploit the advantages of satellite diversity.

2.2.2 GPS Localization and Error Sources

A GNSS receiver on the ground can compute its 3D location, time, and velocity. The key idea is to measure various attributes of the arriving signal (e.g., time of flight, phase, etc.), and then apply statistical algorithms to estimate the errors and ambiguities in measurements. We describe below an overview of the techniques that underpin GPS; other GNSS systems rely on similar techniques.

Pseudorange

When a satellite transmits a signal, it includes the starting time of the transmission (obtained from its atomic clock). The ground receiver records the time of reception also using its less accurate local clock. The time-of-flight (ToF) is the difference between these timestamps. When multiplied by the speed of light, the result gives the *rough* distance to the satellite, called *pseudorange*.

Pseudorange = ToF * (speed of light)

Of course, the measured ToF is inaccurate because the clock of a typical GPS receiver is not synchronized to the GPS satellites. The resulting error can be up to 300 km. In addition, when the GPS signal enters the Earth's atmosphere, it can get delayed due to refractions in the Ionosphere and Troposphere. A signal also passes through a multipath channel, adding more errors. Assuming the true range between a satellite s and receiver i is ρ_i^s, the measured Pseudorange P_i^s, inclusive of all error sources, can be modeled as

$$P_i^s = \rho_i^s + c\,t_i - c\,t^s + A + M_i + \epsilon_i^s \tag{2.1}$$

Here, t_i and t^s are receiver and satellite clock biases, respectively, with respect to true time. A represents the range error due to refractions in the atmosphere. M_i^s denotes Multipath, ϵ_i^s is receiver's hardware noise and c is the speed of light. In today's systems, the satellite clock error t^s is small[1], and $c\,t_i$ proves to be the major source of error. Hence, $c\,t_i$ is modeled as an unknown, and ρ_i^s is written as a function of the *unknown* 3D receiver location ρ_i and the *known* 3D satellite position ρ^s. This results in a total of 4 unknowns. Once we have a lock with 4 satellites, the time bias and the 3D locations can be jointly estimated resulting in a position fix. The ignored error sources, i.e., t^s, A, M, and ϵ, could contribute to an error of 1-4 meters, depending upon the environmental conditions.

GPS receivers on phones and car dashboards use the above techniques. However, higher accuracy applications such as 3D orientation tracking require better performance. To this end, modern GPS research has leveraged the *phase* of the arriving signals, as detailed next.

Carrier Phase

Once a satellite lock is acquired, the phase of the arriving signal, ϕ_i^s, is constantly tracked by a *phase lock loop* (PLL). The true range between the satellite and the receiver, ρ_i^s, can be expressed as a multiplicative factor of wavelength λ.

$$\rho_i^s = \lambda N_i^s + \lambda \phi_i^s \tag{2.2}$$

N_i^s is an unknown integer, meaning that the PLL measurement of ϕ_i^s only captures the fractional

[1] Satellites estimate the errors themselves from mutually exchanged signals as well as from ground sources.

part of the range. However, due to atmospheric effects, multipath, and clock issues, the above equation can be updated:

$$\lambda\phi_i^s = \rho_i^s + c\,t_i - c\,t^s + A + M_i + \epsilon_i - \lambda N_i^s \tag{2.3}$$

Estimating N_i^s is non-trivial and several algorithms have been proposed [43, 44]. We discuss more in Sec. 2.5.1.

One advantage of carrier phase is that *its changes over time* can be tracked reliably by utilizing the doppler shift in the signal [45]. Hence, $\phi_i^s(t_2)$ takes the same mathematical form as Equation 2.3. Thus, if the initial value of N_i^s can somehow be estimated, the tracking thereafter can be good. We now explain how today's systems like Differential GPS (DGPS) with this integer ambiguity, N_i^s, and other error sources.

2.2.3 Computing Differentials

Environmental error sources in Equation 2.3 are correlated over short time periods and within small geographical areas (200 km). Thus, two GPS measurements across time can be subtracted (or differenced) to eliminate some of these factors. Similarly, simultaneous measurements from multiple GPS receivers can also be differenced. Differential GPS (DGPS) [46] performs such operations on *pseudoranges*, while Real-Time Kinematic (RTK) [47] applies differentials to *carrier phase*. For our purpose of precise orientation tracking, the latter is more relevant. To this end, we outline 4 kinds of carrier phase differentials.

(1) Single Differentials across Receivers (SD_{ij})

Consider the carrier phase equations for two GPS receivers i and j from the same satellite s (let us ignore multipath and noise).

$$\lambda\phi_i^s = \rho_i^s + ct_i - ct^s + A^s - \lambda N_i^s \tag{2.4}$$

$$\lambda\phi_j^s = \rho_j^s + ct_j - ct^s + A^s - \lambda N_j^s \tag{2.5}$$

Differencing the above two equations yields the relative position between i and j with fewer error terms. Correlated error sources of atmospheric delays and satellite clock biases disappear.

$$\lambda\Delta\phi_{ij}^s = \Delta\rho_{ij}^s + c\Delta t_{ij} - \lambda\Delta N_{ij}^s \tag{2.6}$$

Figure 2.1 illustrates the scenario. Assuming that the satellite is far away, $\Delta\rho_{ij}^s$ can be approximated as $\rho_{ij}Cos\theta$, where ρ_{ij} is the true relative position between i and j (called **baseline** vector). Replacing $\rho_{ij}Cos\theta$ as a vectorial projection of ρ_{ij} on to the line of sight unit-vector $\hat{l}_s$ of satellite s, we have:

$$\lambda\Delta\phi_{ij}^s = \rho_{ij}.\hat{l}_s + c\Delta t_{ij} - \lambda\Delta N_{ij}^s \tag{2.7}$$

While some errors have disappeared [2], a function of the clock bias errors, $c\Delta t_{ij}$ still remains, motivating the need for double differentials.

[2] We are aware that differentials can amplify noise and multipath, however, since carrier phase noise is in the granularity of few mm [48], we ignore noise in the rest of the chapter. We also assume multipath is not excessive, such as the drone flying low in Manhattan-like areas.

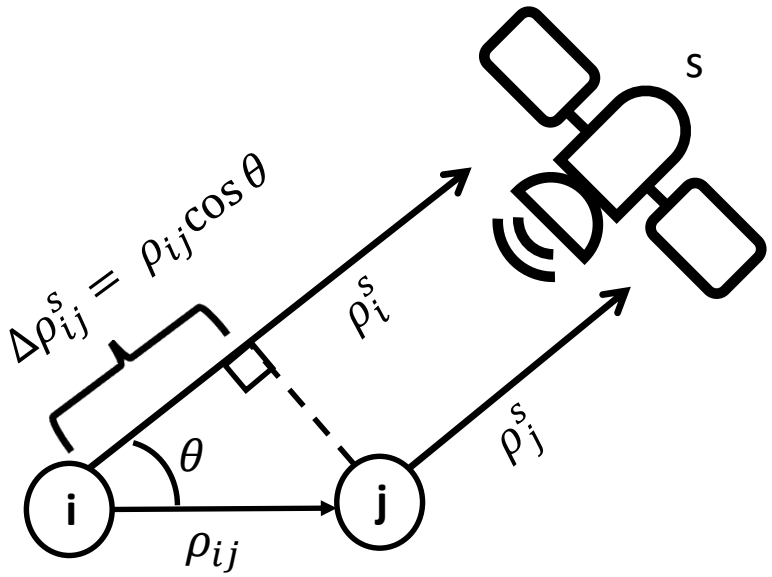

Figure 2.1: $\rho^s_i - \rho^s_j = \Delta\rho^s_{ij} = \rho_{ij}.\hat{l}_s$

(2) Double Differentials across Receivers and Satellites (DD^{sk}_{ij})

Given multiple satellites in range, the GPS ground receivers i and j can perform the same measurements with satellite k. Equation 2.7 can then be rewritten as:

$$\lambda\Delta\phi^k_{ij} = \rho_{ij}.\hat{l}_k + c\Delta t_{ij} - \lambda\Delta N^k_{ij} \tag{2.8}$$

Subtracting the single differential equations 2.7 and 2.8, we have a double differential (DD) as follows:

$$\lambda\nabla\Delta\phi^{sk}_{ij} = \rho_{ij}.(\hat{l}_s - \hat{l}_k) - \lambda\nabla\Delta N^{sk}_{ij} \tag{2.9}$$

The double differential (DD) eliminates the clock biases and the residue is only the integer ambiguity terms. We will discuss the resolution of integer ambiguity in Section 2.5.1, but assuming that the ambiguities are magically fixed, this provides us a reasonably precise estimate of relative positions (called baselines) between receiver pairs. Ignored factors, including multipath,

noise, and antenna phase center errors, add up to a few centimeters of error. Thus, if one of the receiver's absolute position is known in the granularity of millimeters, the absolute position of the other receiver can be estimated precisely as well. Real Time Kinematics (RTK) technology operates exactly as above – *it uses the accurately known location of the reference receiver to calculate the location of the other.* Figure 2.2 shows the relative distance between two receivers placed roughly 45 cm apart. Differencing techniques convincingly outperform naive subtraction of 3D GPS positions.

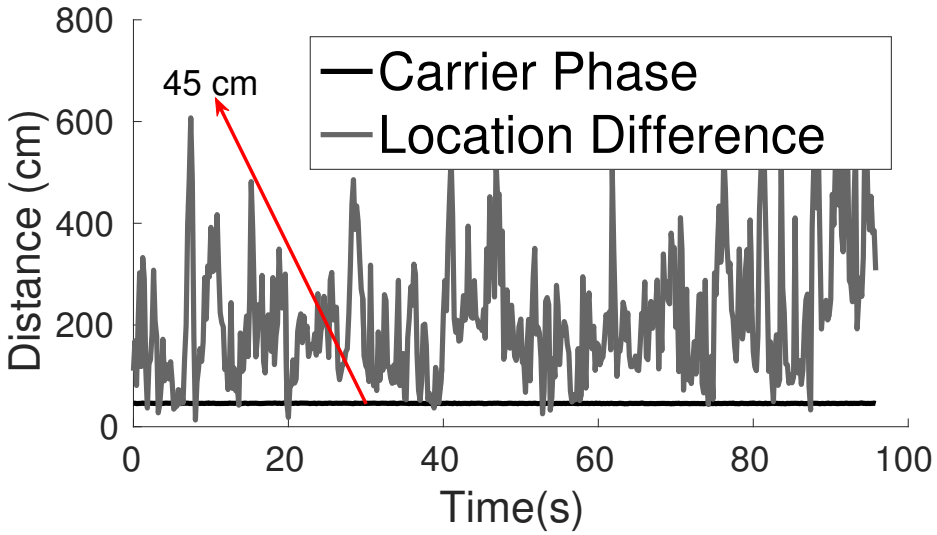

Figure 2.2: Relative positioning using carrier phase and double differentials is an order of magnitude more accurate than naïve location differencing. Here, a 45 cm baseline is correct within a few cm.

(3) Single Differentials across Time ($SD^{t_{12}}$)

Similar to differentials across receivers and satellites, we can also perform differentials across time for the same receiver. This can eliminate the integer ambiguity as follows:

$$\lambda\phi_i^s(t_1) = \rho_i^s(t_1) + ct_i(t_1) - ct^s(t_1) + A^s - \lambda N_i^s(t_1) \tag{2.10}$$

$$\lambda\phi_i^s(t_2) = \rho_i^s(t_2) + ct_i(t_2) - ct^s(t_2) + A^s - \lambda N_i^s(t_2) \tag{2.11}$$

When no cycle slips occur (i.e., $N_i^s(t_1) = N_i^s(t_2)$) and given that the satellite clock bias is well known and hardly changes in small time intervals $[t_1,\ t_2]$, we have

$$\lambda \Delta\phi_i^s(t_{12}) = \Delta\rho_i^s(t_{12}) + c\,t_i(t_{12}) \tag{2.12}$$

Since the distance to the satellite is very large compared to the displacement of the receiver during $[t_1,\ t_2]$, the satellite location can be assumed fixed for this interval, incurring negligible errors with this approximation. The motion of the receiver is then represented in Figure 2.3. As described earlier, $\Delta\rho_i^s(t_{12})$ can now be expressed as the projection of the relative motion vector $\rho_i(t_{12})$ on to the line-of-sight unit vector $\hat{l}_s$ of the satellite. Thus, Equation 2.12 becomes:

$$\lambda \Delta\phi_i^s(t_{12}) = \rho_i(t_{12}).\hat{l}_s + c\,t_i(t_{12}) \tag{2.13}$$

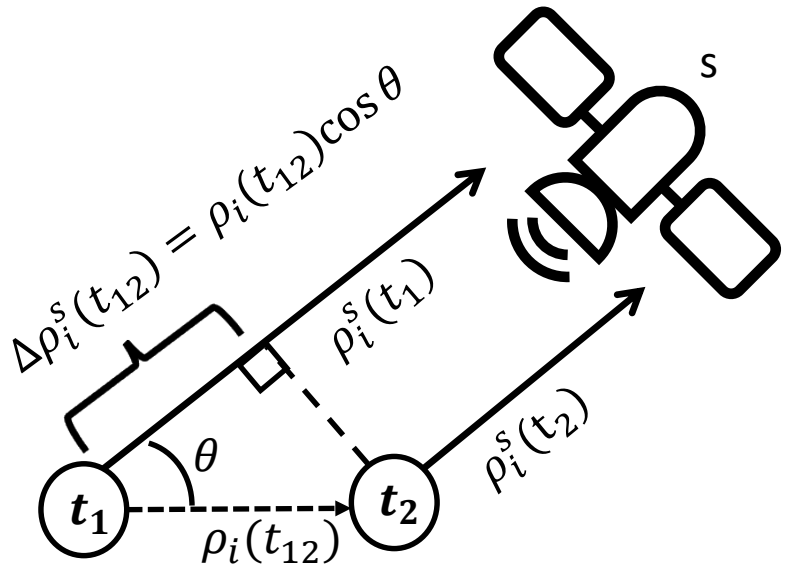

Figure 2.3: $\rho_i^s(t_1) - \rho_j^s(t_2) = \Delta\rho_{ij}^s(t_{12}) = \rho_{ij}^s(t_{12}).\hat{l}_s$

Observe that $\rho_i^s(t_{12})$ (the relative displacement of the receiver during $[t_1,\ t_2]$) and the receiver's clock bias $t_i(t_{12})$ add up to 4 unknowns. Measurements across 4 satellites can help jointly esti-

mate the *relative displacement* and clock bias. This estimated relative displacement is quite accurate since the integer ambiguity N_i^s does not pollute it. We verified this with a static receiver placed on the ground; the estimation resulted in about 1 cm/s of motion, far more accurate than velocity estimations from Doppler shifts.

(4) Double Differentials across Receivers and Time ($DD_{ij}^{t_{12}}$)

Our final double differential combines receivers and time. We compute the single differential between receivers ij from Equation 2.7 but write them for consecutive time points t_1 and t_2.

$$\lambda\Delta\phi_{ij}^s(t_1) = \rho_{ij}(t_1).\hat{l}_s + c\Delta t_{ij}(t_1) + \lambda\Delta N_{ij}^s(t_1) \tag{2.14}$$

$$\lambda\Delta\phi_{ij}^s(t_2) = \rho_{ij}(t_2).\hat{l}_s + c\Delta t_{ij}(t_2) + \lambda\Delta N_{ij}^s(t_2) \tag{2.15}$$

Assuming no cycle slips (we will relax this assumption later), $\Delta N_{ij}^s(t_2) = \Delta N_{ij}^s(t_1)$, hence subtracting the above two equations eliminates integer ambiguity:

$$\lambda\nabla\Delta\phi_{ij}^s(t_{12}) = (\rho_{ij}(t_1) - \rho_{ij}(t_2)).\hat{l}_s + c.\Delta t_{ij}(t_{12}) \tag{2.16}$$

One may physically interpret this equation as the subtraction of two vectors, where the first vector is a drone baseline at time t_1 and the second vector is the same baseline at t_2. In other words, this captures the relative motion of the drone across time.

2.2.4 The Bigger Picture

We take-away 2 key points from the discussion on the differentials:

- The double differentials across receivers and satellites (DD_{ij}^{sk}) yields the drone's baseline vectors at any given time point (Equation 2.9). However, this estimate is still polluted by integer ambiguity.

- The double differential across receivers and time ($DD_{ij}^{t_{12}}$) yields the drone's *relative change* in the baseline vectors during flight (Equation 2.16). Importantly, this relative estimate is free of the integer ambiguities.

Thus, we now have *two* separate estimates of the drone's 3D baseline vectors, each with different error properties. SafetyNet recognizes the opportunity of combining these two noisy estimates to precisely track the drone baselines, ultimately tracking orientation.

2.3 System Model

SafetyNet will model orientation tracking as a state estimation problem, where the state is defined as 3D orientation. We formally define "3D orientation" first and then design the model.

Figure 2.4 pretends 4 GPS receivers have been placed on a drone – their locations denoted as ρ_1, ρ_2, ρ_3 and ρ_4. The baseline vectors joining one of the receivers (say ρ_4) to the others can be defined as $\rho_{ij} = \rho_i - \rho_j$. When the "baselines" are aligned with the North-East reference axes, the baseline matrix B_o can be written as:

$$B_o = [\rho_{41}\ \rho_{42}\ \rho_{43}] \tag{2.17}$$

Assuming that the magnitude of the baseline vectors are d_1 and d_2, we can expand B_o as:

$$B_o = \begin{bmatrix} d_1 & 0 & d_1 \\ 0 & d_2 & d_2 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.18}$$

Figure 2.4 also shows the rotation conventions of the 3 Euler angles – *pitch, roll and yaw.* Applying these rotations on B_o will obviously yield the new baseline matrix, B. For all our results, we will express rotations in terms of the Euler angles (i.e., degrees), which are intuitive

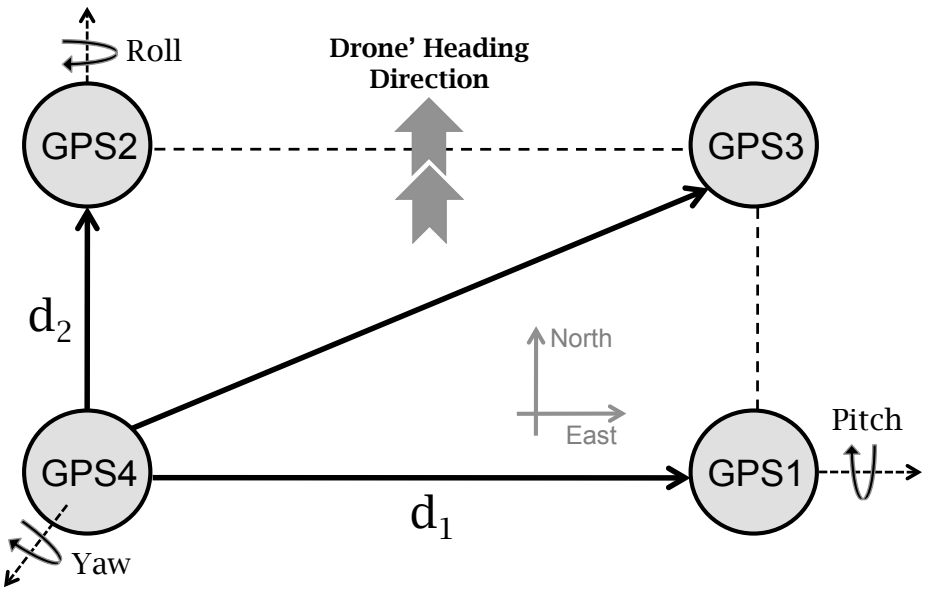

Figure 2.4: The drone baseline vectors ρ_{41} and ρ_{42} aligned with Earth's reference frame.

to understand. However, for the purpose of mathematical efficiency, we will use *quaternion mathematics*, an alternative representation to Euler angles. Briefly, the baselines at an arbitrary orientation, called quaternion q, can be expressed in terms of the initial orientation q_o (aligned with reference axes), as below:

$$\rho_{ij}(q) = A(q)'\rho_{ij}(q_o) \tag{2.19}$$

Here $A(q)$ is the *rotation matrix* associated with the orientation quaternion q. Similarly, extending the effect of rotations to the entire baseline matrix, we have:

$$B(q) = A(q)'B_o \tag{2.20}$$

Effectively, orientation is about estimating the rotation matrix $A(q)$ using projected measurements of $B(q)$ on various satellite directions. Details on conversion between quaternions, Euler angles and rotation matrices can be found in [49]. Regardless of these mathematical conversions, *the core conceptual question still pertains to estimating the matrix B at any given time.*

2.4 System Design: Phase 1

We adopt a Bayesian filtering approach for tracking drone orientation. Figure 2.5 shows the model: (1) A state transition function, derived from the incremental changes in orientation ($DD_{ij}^{t_{12}}$), models the next state of the drone. Recall that these changes are affected by the hardware noise and multipath errors. (2) A measurement function, (DD_{ij}^{sk}), reflects the absolute orientation of the drone at any given time. Of course, this measurement is polluted by integer ambiguity. We adopt a Kalman Filter to combine the uncertainties from the transition and measurement functions, and track the most likely state of the system through time. We describe this basic design first. Then we focus on resolving the error sources (such as integer ambiguity, cycle slips, and missing data), and redesign the framework to accommodate these optimizations. Our final design is an "adjusted" particle filter algorithm that tracks drone orientation with consistent accuracy.

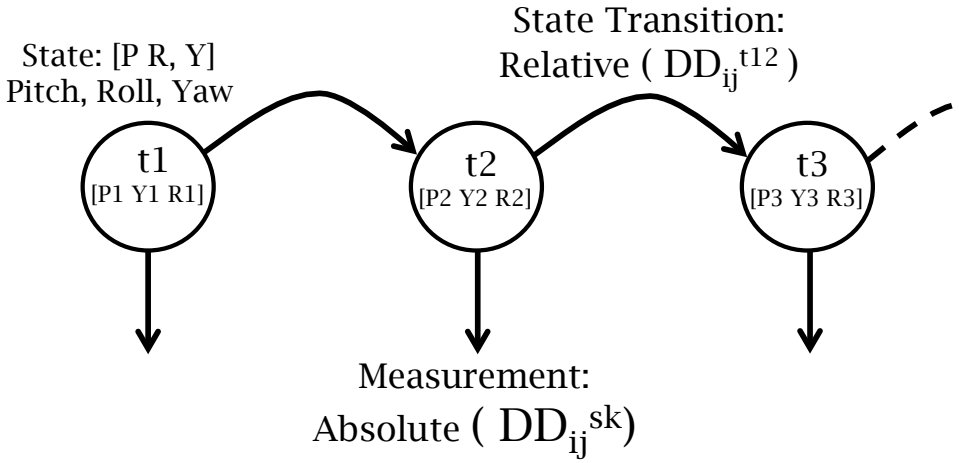

Figure 2.5: Bayesian filtering approach to tracking orientation state over time.

2.4.1 State Transition Model

The relative baseline changes over time are directly obtained from Equation 2.16, copied for convenience:

$$\lambda \nabla \Delta \phi_{ij}^{s}(t_{12}) = (\rho_{ij}(t_1) - \rho_{ij}(t_2)).\hat{l}_s + c.\Delta t_{ij}(t_{12}) \tag{2.21}$$

Omitting details, we rewrite with quaternions:

$$\lambda \nabla \Delta \phi_{ij}^{s}(t_{12}) = \rho_{ij}(q_o)\lfloor A(q_1).\hat{l}_{s\times} \rfloor \delta\theta + c.\Delta t_{ij}(t_{12}) \tag{2.22}$$

where, q_1 is the orientation quaternion at time t_1, $\delta\theta$ is the rotation vector [50] associated with quaternion δq. $\lfloor \ _\times \rfloor$ is the vector cross operator [51]. We now solve Equation 2.22 for various satellites s and GPS receiver-pairs ij using least-squares estimation. The result yields an estimate of the rotation vector $\delta\theta$ (hence δq) between two time points t_1 and t_2. We can thus estimate the new orientation quaternion q_2 as:

$$q_2 = \delta q \otimes q_1 \tag{2.23}$$

Here, $\otimes$ is the quaternion multiplication operator. When translated back to Euler angles, the result is the relative orientation change – pitch, yaw, and roll – from one state to the next. As mentioned earlier, this estimate is polluted by hardware noise and multipath.

2.4.2 Absolute Orientation Measurement

Equation 2.9 showed that double differentials across receivers and satellites (DD_{ij}^{sk}) are estimates of the *absolute* baseline vectors of the drone. While they cancel out clock bias errors,

they leave the integer ambiguities as follows:

$$\lambda \nabla \Delta \phi_{ij}^{sk} = \rho_{ij}.(\hat{l}_s - \hat{l}_k) + \lambda \nabla \Delta N_{ij}^{sk} \tag{2.24}$$

Translating to quaternions again, using very similar conventions as described earlier, we have:

$$\begin{aligned} \lambda \nabla \Delta \phi_{ij}^{sk} - \rho_{ij}(q_o).A(q_n).(\hat{l}_s - \hat{l}_k) = \\ \rho_{i,j}(q_o).\lfloor A(q_n).(\hat{l}_s - \hat{l}_k)_{\times} \rfloor \delta\theta + \lambda.\nabla \Delta N_{ij}^{sk} \end{aligned} \tag{2.25}$$

We develop techniques for resolving integer ambiguities in Section 2.5.1. For now, let's assume they are resolved – then, we are left with a set of linear equations over different satellite pairs sk and baselines ij. By solving them using standard Least Squares Estimation (LSE), the rotation vector $\delta\theta$ and associated quaternion δq can be obtained. Hence the orientation q is.

$$q = \delta q \otimes q_n \tag{2.26}$$

Here q_n is an initial orientation estimate for the purposes of linearization of Equation 2.25 (usually comes from the transition model). Note that this estimate q is absolute in the earth's reference frame, since the satellite locations sk are both known in that reference frame.

2.4.3 Applying Extended Kalman Filter (EKF)

Kalman filter is an estimation algorithm to systematically combine erroneous observations about a system state. In our case, since the transition function is intrinsically non-linear (because of the conversion from rotation matrices to quaternions), we make linearized approximations using Extended Kalman Filter (EKF) [52]. We briefly sketch the methodology here.

Transition: The state transition function in Equation 2.22 can be re-written in a generic non linear model as:

$$\hat{q_{k+1}} = f(q_k) + w_k \tag{2.27}$$

Here, k and $k+1$ are two successive time points, f is of the (non-linear) form:

$$f = \delta q \otimes q_k \tag{2.28}$$

and w_k is the estimated process noise derived from the GPS signal SNR. δq is the quaternion of $\delta\theta$ in Equation 2.22. Now, linearizing Equation 2.27 around the estimated rotation vector associated with quaternion q_k, we have:

$$\delta\theta_{k+1} = F.\delta\theta_k + w_k \tag{2.29}$$

where F is given by $\frac{\partial f}{\partial \theta}|_{\hat{\theta_{k+1}}}$

Measurement: As with the transition function above, the measurement function can be Equation 2.25, written in the following form:

$$y - y_o = H.\delta\theta_{k+1} + v_t \tag{2.30}$$

where k used for both satellite and sample index. v_t is the measurement noise derived from GPS SNR.

$$y = \lambda\Delta\phi_{ij}^{sk} - \lambda.\Delta N_{ij}^{sk} \tag{2.31}$$

$$y_o = \rho_{ij}(q_o).A(\hat{q_{k+1}}).(\hat{l_s} - \hat{l_k}) \tag{2.32}$$

$$H = \rho_{ij}(q_o).\lfloor A(\hat{q_{k+1}}).(\hat{l_s} - \hat{l_k})_\times \rfloor \tag{2.33}$$

Equations. 2.29 and 2.30 can now be combined using an Extended Kalman Filter for a refined estimate of $\delta\theta_{k+1}$. Its associated quaternion δq_{k+1} can then be used to estimate the most likely orientation quaternion q_{k+1}:

$$q_{k+1} = \delta q_{k+1} \otimes \hat{q_{k+1}} \tag{2.34}$$

While the noise terms of Transition and Measurement models are not completely independent, we introduce Particle Filters later (Section 2.5.3) to lift the Gaussian assumption of Kalman Filters.

2.5 Resolving Ambiguities: Phase 2

The above EKF has been designed under the convenient assumption that integer ambiguity and cycle slips have been resolved. However, this resolution is challenging, as evident from the numerous projects written on this topic [53–56]. Since these error sources seriously affect drone orientation, we comprehensively address them next.

2.5.1 Resolving Integer Ambiguity

Recall from Equation 2.2 that the Phase Lock Loop (PLL) only measures the fractional part of the range, leaving an unknown of λN. Once the drone is flying, additional multiples of wavelengths can also accumulate, called *cycle slips*. We first describe ways to mitigate the initial λN and discuss cycle slips thereafter.

Note that we do not need to estimate N since we are not computing the actual range (or location) of the GPS receiver. Since we only care about orientation, we have been operating in the space of differentials. Thus, when we compute $N_{ij}^{s} = N_{i}^{s} - N_{j}^{s}$, followed by $N_{ij}^{sk} = N_{ij}^{s} - N_{ij}^{k}$, we recognize that N_{ij}^{sk} are also integers. Our goal is to estimate these N_{ij}^{sk} integers, for all

combinations of ij and sk.

Our core intuition is to compute a Cos function on the value of $2\pi N_{ij}^{sk}$; since N_{ij}^{sk} should be an integer, $Cos(2\pi N_{ij}^{sk})$ should equal 1. Now, let's say this Cos function is summed up over all the possible values of N_{ij}^{sk}, then we have $\sum_{ij,sk} Cos(2\pi N_{ij}^{sk})$. If there are ψ possible tuples of $< ij, sk >$, then the summation should add up to ψ. Now, this summation can be rewritten as a function of orientation q as follows:

$$cos(2\pi N_{ij}^{sk}) = cos(2\pi \frac{\lambda\phi_{ij}^{sk} - \rho_{ij}(q_o).A(q).(\hat{l_s} - \hat{l_k})}{\lambda}) \tag{2.35}$$

Here N_{ij}^{sk} is derived from a combination of Equations 2.9 and 2.19. Then,

$$M(q) = \sum_{ij,sk} cos(2\pi N_{ij}^{sk}) \tag{2.36}$$

In an ideal case, the correct q should make the RHS of Equation 2.36 equal to ψ. However, given that phase ϕ_{ij}^{sk} in Equation 2.35 has errors, for a given q, the corresponding values of N_{ij}^{sk} are not all 1. When these N_{ij}^{sk} are plugged into Equation 2.36, the value is less than ψ. Hence, we search across all values of q at the granularity of 5 degrees and pick a q_{coarse}, such that

$$M(q_{coarse}) = \max_{q} M(q) \tag{2.37}$$

5 degrees is not an arbitrary design choice. Rather, 5 degree is a function of GPS signal wavelength and the diagonal of the drone, together ensuring that the estimated values of N_{ij}^{sk} do not over or undershoot an integer by more than 0.2 with high probability. So long the offset is less than 0.5, the actual N_{ij}^{sk} values can be estimated by rounding off. The equation below shows

our final estimation of integer ambiguity from q_{coarse}.

$$\hat{N_{ij}^{sk}} = \lfloor \frac{\lambda \phi_{ij}^{sk} - \rho_{ij}(q_o).A(q_{coarse}).(\hat{l_s} - \hat{l_k})}{\lambda} + 0.5 \rfloor \tag{2.38}$$

Our results show that when the drone is static and ready to take off, the estimates of N_{ij}^{sk} are correct (also, the Cos function is close to 1). However, once the drone starts flying, and especially after sharp maneuvers, integer ambiguity again builds up. This is because poor SNR, exacerbated by sudden phase changes in the BPSK GPS signal, causes the PLL to lose lock. The result is both full and half cycle slips[3]. Now, even if q_{coarse} correctly estimates a large fraction of N_{ij}^{sk}, the small erroneous fraction influences subsequent estimation and the orientation diverges over time. Moreover, half cycles also increase the search space by $8x$ since the search is 3-dimensional (yaw, pitch, roll). Thus, coarse grained attitude, q_{coarse}, can no longer be used for consistent tracking.

Fig.2.6 compares the *cycle slip percentage* for static and flying drones – we use the view from the drone camera as the ground truth [57] (described later in Section 5.4). Evidently, cycle slips occur much more frequently (almost once every second) when the drone is flying. This means that even our assumption in Equation 2.12 – that $N_i^s(t_2) = N_i^s(t_1)$ – is not necessarily true, and can drastically pollute estimates. To this end, SafetyNet approximates q_{coarse} for Equation 2.38 from the estimate of the State Transition Model in Section 2.4.1. While this is not perfect, the Outlier detection and Particle Filter techniques discussed next will help absorb most errors.

2.5.2 Cycle Slip Detection

Needless to say, our cycle slip detection scheme has to be quick (to be able to run every few seconds). Towards this goal, we form temporal differential measurements from multiple

[3]Half cycle slips occur because some PLLs square the signal to remove BPSK messages; squaring makes the correlator match the original signal twice within a signal period, indicating phase change. However, this squaring effectively halves the wavelength, meaning that N_{ij}^{sk} is now multiples of 0.5.

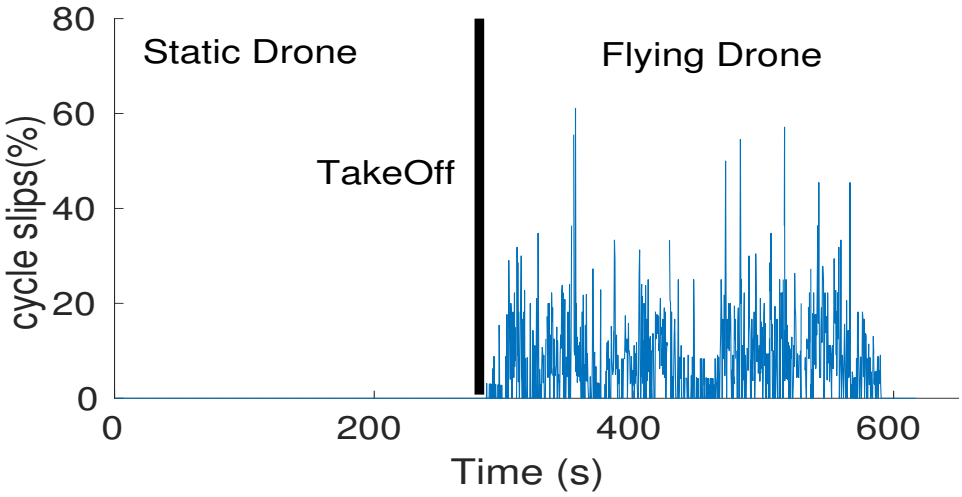

Figure 2.6: Cycle slips occur more often during flight.

satellites as shown in Equation 2.13. After compensating for known satellite biases $t^s(t_{12})$, we are left with four unknowns – 3D change in position $\rho_i(t_{12})$ and clock bias difference $t_i(t_{12})$. Solving these using *Robust Least Squares (RLS)* [58] will typically isolate the outliers, with Error Residuals for the inliers being less than $2cm$. Our transition model (Section 2.4.1) incorporates corrections from this module. While there is a possibility that RLS may not converge, we use GLONASS satellites to boost the redundancy.

Unlike GPS satellites, GLONASS uses FDMA, i.e., satellites transmit in different frequencies. Since the wavelengths are now λ^s and λ^k from satellites s and k, the integer ambiguity terms do not group themselves into a single integer in the DD_{ij}^{sk} double differentials. Hence, we use GLONASS only for $DD_{ij}^{t_{12}}$, since the signals from the same satellite cancel across time, t_{12}. However, when even GLONASS+GPS fails (perhaps because of too few satellites under poor weather), we resort to Particle Filters (Section 2.5.3).

2.5.3 Unified Particle Filter Framework

The net result of all the above techniques can be summarized as follows: *the orientation q, and hence the values of N_{ij}^{sk}, are mostly correct, and the Kalman filter can track it well. However, occasionally q is incorrect, and this error accumulates over time causing complete divergence.* Figure 2.7 illustrates an example of the drift with Kalman Filters (KF). We bring Particle Filters to better handle these cases since multiple particles (i.e., orientation states) can be propagated through time, while relaxing linearity and Gaussian assumptions required for KF. Thus, SafetyNet adopts a hybrid approach, i.e., the system uses Kalman Filters in general but invokes a particle filter when:

- Orientation estimates have poor confidence based on the summed Cos metric in Equation 2.36.
- Missing data in one or more of the receivers derails the state transition function.

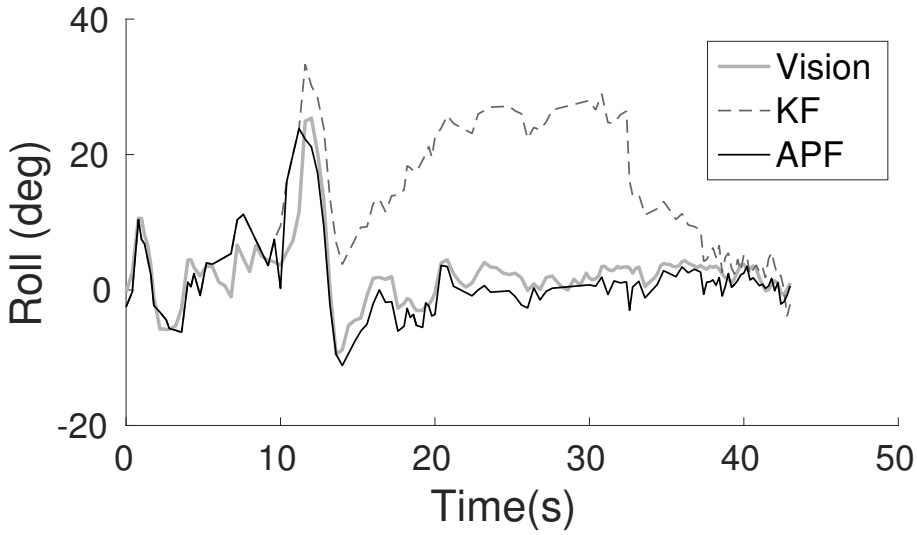

Figure 2.7: KF based orientation drifts away due to poor SNR or missing data.

Figure 2.8 summarizes the flow of events. When the confidence on q (based on the Cos metric) is high, SafetyNet continues to run the Kalman filter. When low, we select K values of orientation q – the values whose corresponding Cos metric ranks in the top-K. We initialize K particles

at each of these orientation states and update each particle based on the transition and measurements functions. These particles are essentially "trackers" of different integer ambiguities, and we propagate them until the Cos metric for one becomes high. Now, if the particles are too close to each other – within 3° – we merge these particles into one. If a particle exhibits extremely low weight, we also remove that particle. If merging or removal leaves one particle, SafetyNet switches back to Kalman filtering again. Else, the confidence function is computed again. If the confidence function is low, more particles are added and resampled – resampling is performed such that the resulting particles are an equal mix of the (highest weighted) current and new particles. However, if the confidence is high, adding and resampling is not necessary. The process repeats until the system converges to a single particle.

2.5.4 Adjusted Particle Filter (APF) Design

We propose an improvement to the particle filter design. The opportunity arises from the observation that in the conventional particle filter, a particle's weight is typically proportional to its likelihood function (i.e., proximity to the true state). Thus, when the system resamples, the concentration of particles are proportionally greater at the higher weight particles. Over time, the hope is that one of the particles would converge to the correct state. However, we ask: *why not move the particle to a state that maximizes the likelihood.* In fact, given that the error variances of the transition and measurement functions are known, the best estimate can be computed as a combination of the two. We could move each particle to the "best" state in its neighborhood, and then perform the resampling step.

Figure 2.9 illustrates the idea for a case of three particles. The middle particle is propagated to time $time_1$ using the transition model $DD_{ij}^{t_{12}}$. Similarly, the measurement function DD_{ij}^{sk} produces an estimate of the new state. The transitioned and measured states along with their error distributions are depicted. The two estimates are combined using a Kalman filter like approach, and the particle "adjusted" to this best state. This particle is then propagated further

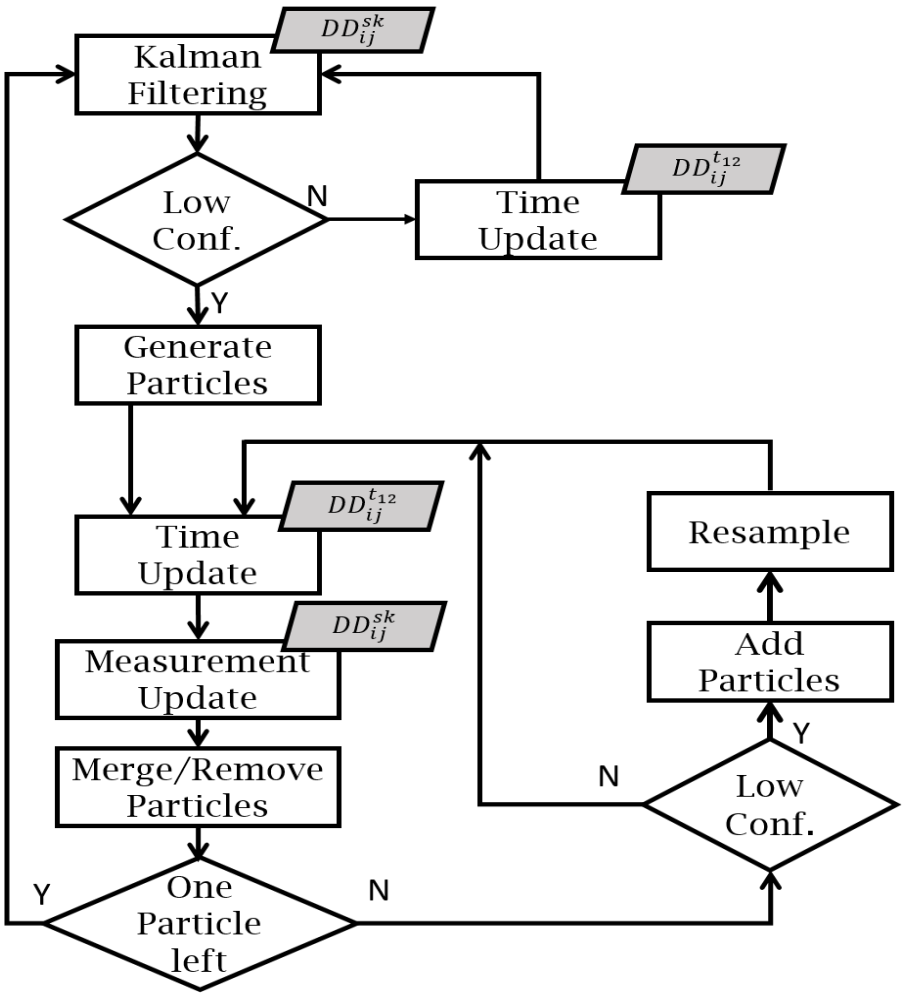

Figure 2.8: Flow chart of unified particle filter

to time $time_2$, which now has a smaller error variance.

We note that each particle essentially tracks a particular N_{ij}^{sk} integer ambiguity vectors. As a result, every carrier-phase measurement yields distinct orientation q for each of these vectors. The benefits arise because: (1) For the correct integer ambiguity vector, the error properties of the measurement function becomes Gaussian. Hence, the combining process indeed is like a Kalman filter, leading to faster convergence. (2) For incorrect integer ambiguity vectors, the gaussian error properties will not hold. Moreover, the transitioned particle will be adjusted towards a wrong state because of incorrect integer ambiguity resolution. Thus, the net outcome is

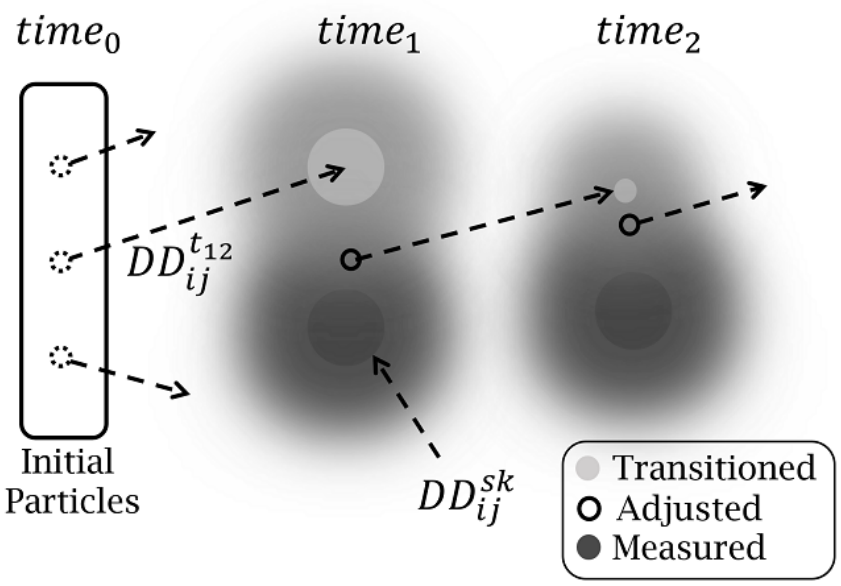

Figure 2.9: Adjusted Particle Filter (APF) results in faster reliable convergence even with few particles.

faster convergence for the correct ambiguities, while disappearing particles for incorrect ambiguities. Figure 2.7 shows the efficacy of the *Adjusted Particle Filter* (APF) even when KF diverges. The next section presents more detailed results.

2.6 Evaluation

Our evaluation will comprehensively compare SafetyNet's orientation accuracy against two reference points: (1) the real IMU of a professional-grade drone (using algorithms based on [59]) and (2) high-precision estimates of ground truth through computer vision, which is orthogonal to both inertial and GPS-based sensory modalities. All results are from 11 flight sessions of ≈ 6 minutes each, across a diverse set of weather and satellite conditions; we will also report 95^{th} percentile performance to capture robustness. We aim to answer the following critical ques-

tions.

- What is SafetyNet's accuracy in estimating a drone's pitch, roll, and yaw during flight? *(Figures 2.11, 2.12a)*
- Is SafetyNet's accuracy comparable to IMU, relative to ground truth from computer vision? *(Figure 2.12(b,c))*
- Is SafetyNet robust to weather impairing satellite visibility? *(Figures 2.13, 2.14)*
- Is SafetyNet robust under aggressive maneuvers? *(Figures 2.11, 2.15)*
- Is SafetyNet computationally practical for embedded, real-time applications? *(Figure 2.16)*
- Which modules of SafetyNet provide the largest performance gains? *(Figure 2.17)*

Our results show that SafetyNet demonstrates consistently dependable performance in each measure, and that all of SafetyNet's modules are critical to the net outcome. We begin by describing the platform and methodology.

2.6.1 Experimental Platform and Methodology

We conduct all experimentation using a 3DR X-8 octocopter (8-rotor), pictured in Figure 2.10. "X" implies 4 arms with unequal spacing – yielding greater stability of roll angle versus pitch (recall that pitch is the dominant motion when an airplane takes off or lands, roll is dominant when the plane makes a left/right turn). Two rotors attach to each arm: one above, one below. To the X-8, we mounted 4 u-Blox NEO-M8T multi-GNSS receivers, adjacent to the top motor of each arm. We also mounted a Raspberry Pi 2 near the drone's center of mass – the GPS receivers transfer the data to the Pi via USB at 5 Hz. IMU data at 10 Hz is recovered as binary logs from the X-8's USB interface. [4]

[4] Experiments were conducted as per FAA regulations. Operator trained by a general aviation pilot, over relatively vacant fields, with flight height not exceeding 45m.

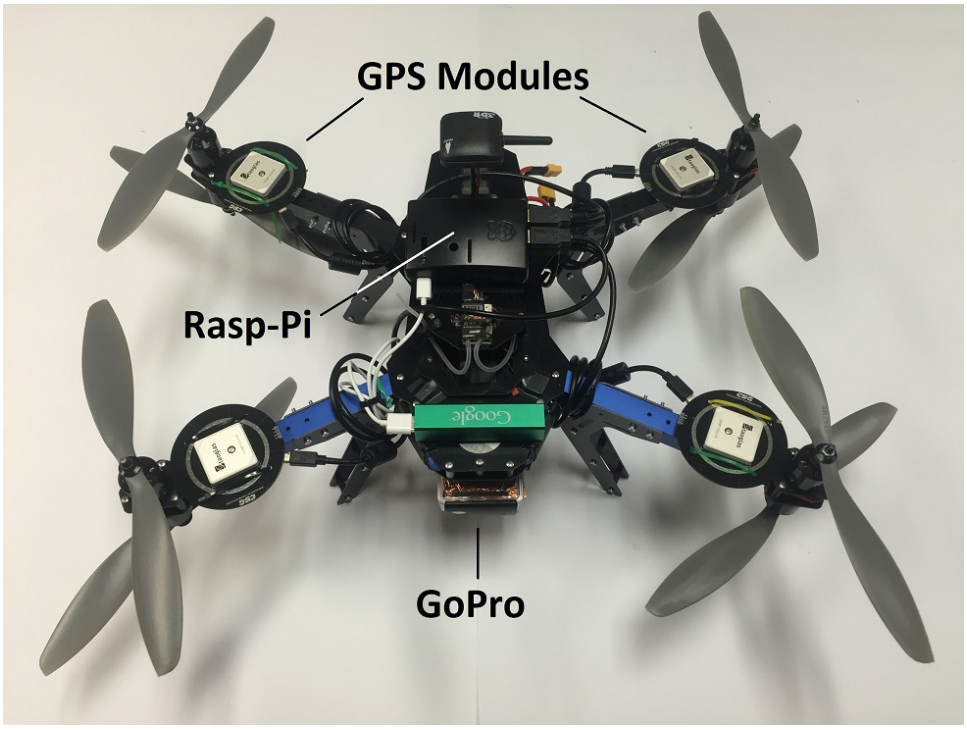

Figure 2.10: Four off-the-shelf GPS modules, a Raspberry Pi, and a GoPro on an X-8 Octocopter

A GoPro camera, affixed to the underbelly of the airframe, points vertically downwards. The GoPro captures video at 30 fps with a fast shutter (to minimize motion blur and rolling shutter effects). We use `ffmpeg` to sample the video into stills at 7.5 fps (every fourth frame). We post-process the images using *Pix4D*: advanced commercial photogrammetry software that uses *structure from motion* (SfM) to perform 3D registration of each frame. From Pix4D's outputs, we can recover a high-precision estimate of drone attitude (accuracy $\approx 0.05^\circ$ [57, 60]). post-process the images using *Pix4D*: advanced commercial photogrammetry software that uses *structure from motion* (SfM) to perform 3D registration of each frame. Note: structure from motion would not be practical as a realtime IMU replacement – each 5-7 minute flight requires several hours of processing on a server of 16 CPU cores, 32 GB RAM, and CUDA on a high-end NVIDIA GRID K2 GPU.

2.6.2 Data Alignment for Comparison

SafetyNet and IMU data have precise GPS timestamps but the GoPro does not – this makes precise comparison difficult. Therefore, at the beginning/end of each flight, we point the GoPro towards an Android phone displaying the current GPS time. Once the video frame is synchronized with the GPS time, we extrapolate and time stamp every frame – this is possible because we extensively verified that the inter-frame spacing of the GoPro is uniform (33.33 ms at 30 fps). Still, residual error remains from unpredictable OS delay in the Android application displaying the current time. We align the yaw *angle sequence* between SafetyNet and vision. We also perform this synchronization procedure with IMU to eliminate residual misalignment. Finally, we align the GoPro and GPS X-Y axes by finding the appropriate rotation matrix that eliminates offsets in yaw, pitch, and roll. We now present results.

2.6.3 Performance Results

(1) Overall Tracking Accuracy: Figure 2.11 depicts SafetyNet closely tracking (a) pitch, (b) roll, and (c) yaw against Pix4D/vision. Figure 2.12(a) aggregates 60 minutes of flying data: median error of 2.07° for pitch, 1.38° for roll, and 0.61° for yaw. Higher accuracy for roll versus pitch is attributable to our X-shaped drone: the x-axis baseline is 46% longer than the y-axis. Accordingly, roll is 49% more accurate. As expected, yaw accuracy is higher than pitch or roll as it is relatively less subject to GPS dilution of precision from satellite geometry.

(2) Comparison with IMU: Figure 2.12(b and c) compares the accuracy of SafetyNet with IMU, treating Pix4D/vision results as ground truth. SafetyNet is relatively less accurate ($\approx$ 30% worse) than IMU in pitch and roll. However, SafetyNet comprehensively outperforms IMU in yaw angle accuracy ($\approx$ 300% more accurate / one-quarter error). Inferior yaw accuracy of IMU, relative to pitch and roll, is expected due to reliance on magnetometer versus the accelerometer gravity vector. Again, SafetyNet's superior yaw performance is expected. We are encouraged

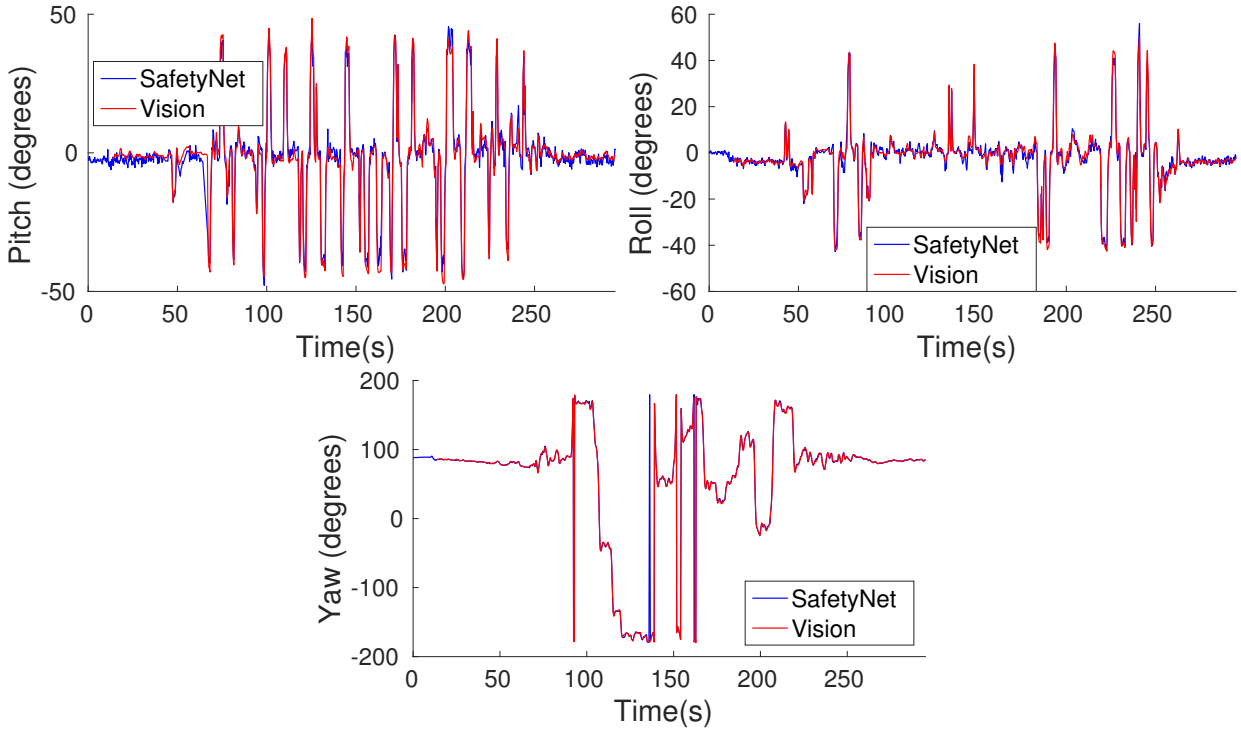

Figure 2.11: Time trace of SafetyNet orientation compared with Vision (a) Pitch (b) Roll and (c) Yaw. SafetyNet can track highly aggressive maneuvers well

that SafetyNet's pitch/roll accuracy is acceptable as a plausible IMU substitute.

(3) Robustness to Weather: Figure 2.13 plots median accuracy over flights/flying conditions. Error bars denote 5th and 95th percentiles. Flights 7-11 were conducted in fog. Despite fewer detectable satellites, SafetyNet performance remains consistent and robust. To emulate extreme weather scenarios, Figure 2.14 shows accuracy with varied fractions of satellites artificially removed from processing. The medians and 75th percentiles are mostly robust to satellite skipping. As evident from the 95th percentile line, worst case errors start increasing at 40%. Having fully exploited the u-Blox NEO-M8T multi-GNSS capabilities, SafetyNet maintains high satellite count, increasing reliability.

(4) Robustness to Maneuvers: Figures 2.11 and 2.15 show that SafetyNet tracks aggressive

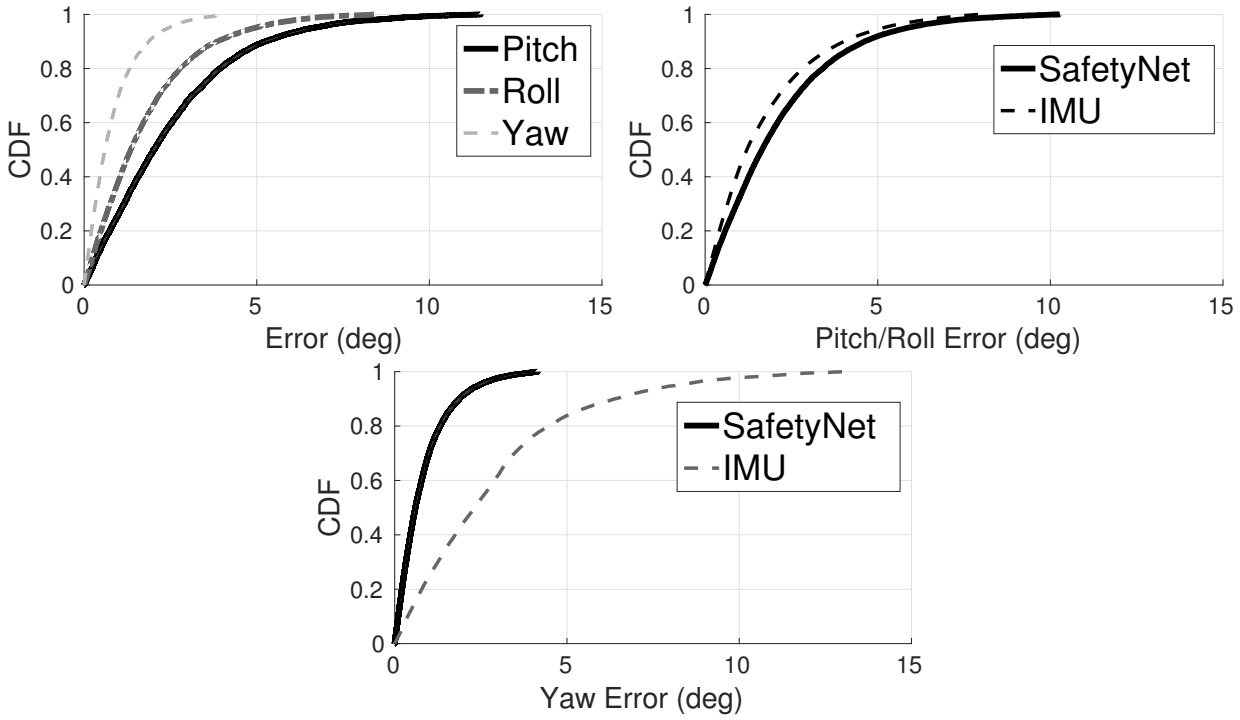

Figure 2.12: (a) CDF of errors for an hour of aggressive flight data shows the effectiveness of SafetyNet. (b) Comparison of SafetyNet's Pitch/Roll against IMU's. (c) Comparison of Yaw against IMU. SafetyNet outperforms IMU in Yaw by 4x while Pitch/Roll is 30% worse.

maneuvers. Median error along with 5th and 95th percentiles is plotted as a function of the pitch/roll/yaw rates. Tracking of extreme motions (120+° per second) becomes challenging because of satellite blockage, cycle slips, and missing samples. Yet, SafetyNet's Particle Filter Framework is robust enough to even the most challenging motions.

(6) Implementability: Figure 2.16 plots the 95%-ile error in pitch, roll and yaw as a function of number of particles. There are two take-aways: (1) SafetyNet's particle filter dramatically reduces error and (2) only few particles are required – the introduction of only a second particle yields a 300% improvement for pitch/roll. We need not trade away the real time implementability of Kalman filters to gain accuracy. SafetyNet can be implemented in realtime on embedded hardware.

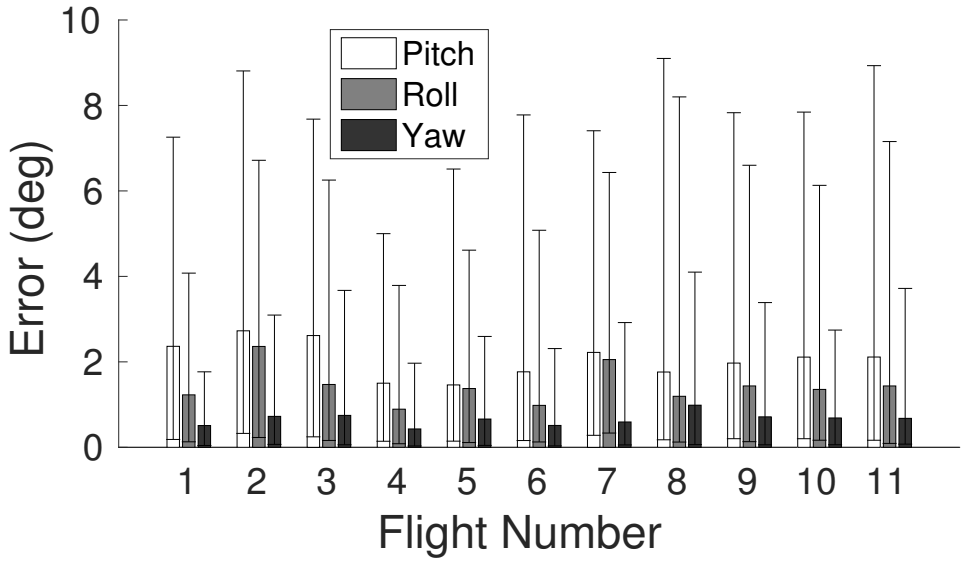

Figure 2.13: Accuracy in SafetyNet is consistent over different flights under diverse satellite visibility

(7) Gain by Module: Figure 2.17 isolates error reduction from each SafetyNet module: Integer Ambiguity resolution (IA), Outlier elimination (OT), Kalman Filter (KF), Simple Particle Filter (PF), Adjusted Particle Filter (APF) and multi-GNSS (GS). IA suffers from cycle slips that OT can eliminate (7x gain in median and 2.5x at 95th). After OT, some wrong integer ambiguities may remain causing KF drift with slow convergence back to a correct state. APF enables rapid convergence (100% median accuracy gains over both PF and KF/ error reduced by 50%). Integrating GLONASS and SBAS [61] improves median performance and dramatically reduces worst case errors (33% median gain, 6x gain at 95th).

2.7 Related Work

Relative GPS: Today's state of the art in GPS relative localization are [32, 33], which outperform DGPS [46] and RTK [47]. Among these, *APT* in [33] is closer to iMob and also uses carrier phases combined with double differentials; they achieve sub-meter level accuracy. iMob's differences include: (1) A completely different mathematical model compared to the modeling strategy

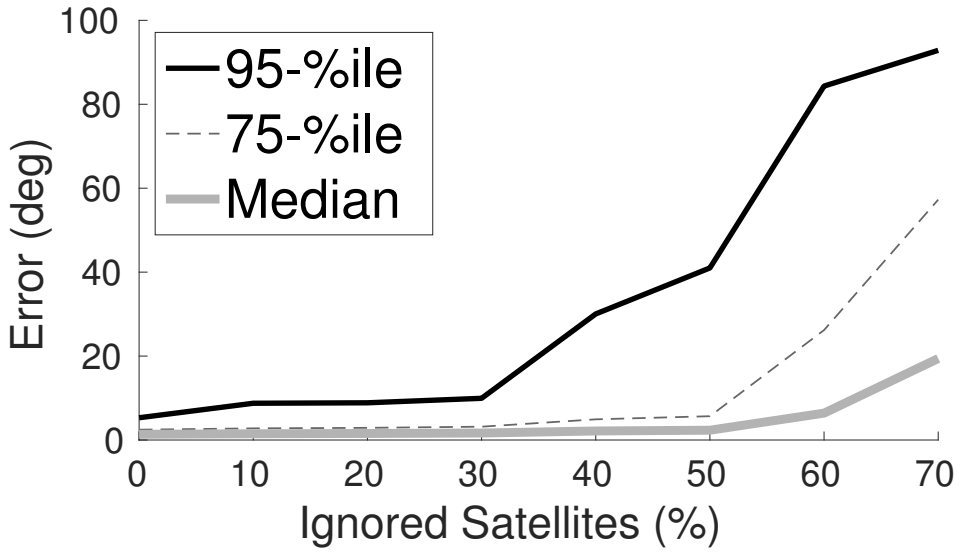

Figure 2.14: Number of outliers can increase with fewer satellite availability, but iMob maximally exploits multi GNSS satellite availability

used for relative localization in *APT*. (2) A notion of "soft decoding" where multiple uncertainties are propagated from the Particle Filter's Cos metric in iMob. iMob also picks new particles from the *angular domain* resulting in dramatic reduction of search space, whereas the search cube based recalibration in *APT* is highly complex. This also facilitates meeting the high accuracy requirement in iMob under a more challenging cycle slip rate. (3) iMob incorporates the inclusion of GLONASS, which is non-trivial due to satellites operating on different frequencies (FDMA). (4) iMob addresses half-cycle slips by resolving integer ambiguities in steps of 0.5 – critical here, given the higher accuracy requirement (1-2 cm) compared to *APT*.

Orientation/Velocity from Multi-GPS: While several works have considered GPS for orientation of spacecraft, full-scale aircraft, or ships [9, 10, 62], none address the dynamic errors for a small drone. While [34] controls a model helicopter using multi-GPS, the motions are highly constrained (slow changes less than 10°). iMob is unique in addressing dynamic integer ambiguities across an aggressive flight (angular changes up to $150°/s$). Solutions using single receiver, multi-antenna GPS [63] will also benefit from iMob for dynamic error handling. Work

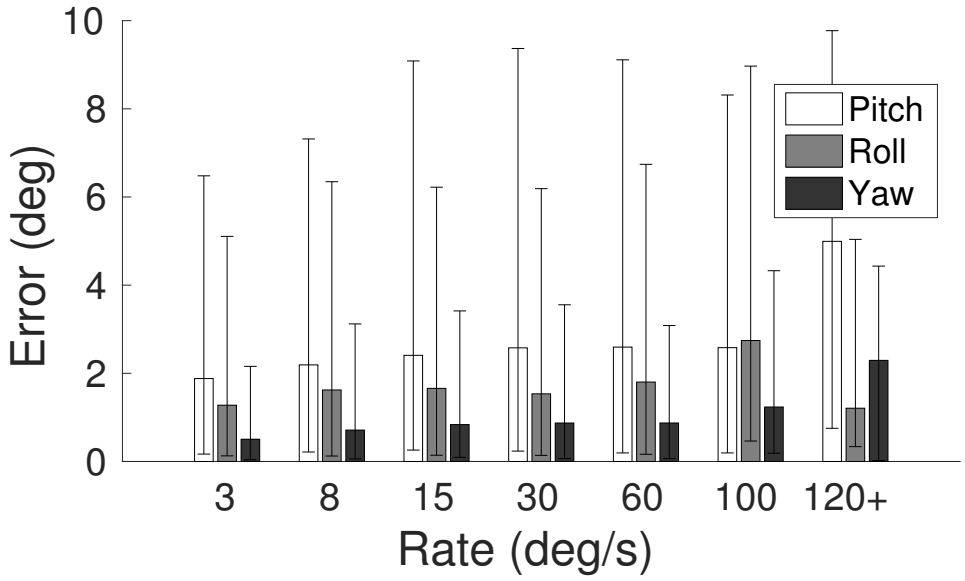

Figure 2.15: SafetyNet's accuracy degrades gracefully with aggressiveness

in [64] uses a moving antenna for orientation estimates. While interesting, the authors note that the motion has to be slow enough to not break GPS locks, hence it is not possible to adopt such solutions for high dynamic flights. Angular velocity determination using multi-GPS is explored in [65, 66]. However, iMob is unique in integrating angular velocity with double differentials (across time and satellites) into a Particle Filtering framework.

Multi-GNSS Fusion: Several works have attempted to fuse GPS with GLONASS satellite systems for enhanced relative positioning [67–69]. However, each use pseudorange data only or assume a specialized multi-antenna GPS/GLONASS receiver with a single clock. iMob exploits time-differenced carrier phase measures to leverage GLONASS data in double differentials with off-the-shelf multi-GNSS receivers.

Reliability for IMU: Drift on gyroscopes [6, 7, 70] and interferences to magnetic compasses [71–73] is well known in smartphone applications and many of these researchers have proposed application specific inferencing techniques to handle them. While the same error

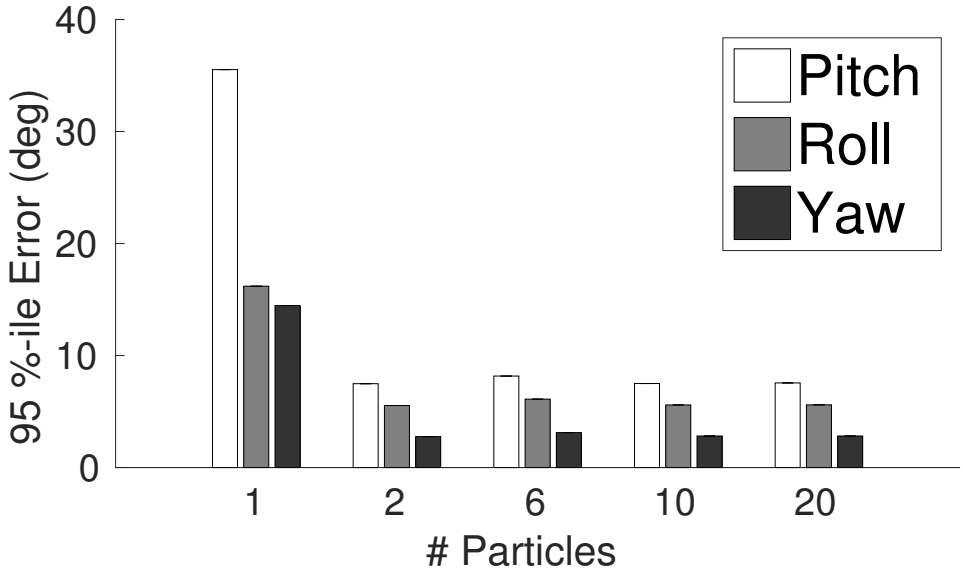

Figure 2.16: SafetyNet can eliminate outliers with very few particles

properties can extend to drones as well, the results could be catastrophic and error handling is very critical. Work in [27] proposes inferencing techniques for correcting IMU errors from engine vibrations for drones. Similarly, [28] models magnetic sources of interferences to nullify the effect on IMU. Work in [30] models above similar noisy sources affecting IMU updates into a sophisticated Kalman filter. In contrast to these works iMob offers a completely orthogonal solution relying only on GNSS receivers.

IMU/GPS fusion: Fusion of IMU sensors like accelerometers, gyroscopes, magnetometer and GPS has been extensively used for orientation tracking [8]. Extended Kalman Filters [52, 59, 74, 75] have been used for handling non-linearity. UKF, Sigma Point Filters and Particle filters have also been used for better accuracy with higher complexity [76–79]. While complementary to iMob in reducing orientation uncertainty, these do not address IMU failure.

Orientation from Vision: Pitch/roll angles can be estimated by tracking the horizon [80, 81]. Optical flow [82], stereo vision [11] and feature tracking [12, 83–85] have been proposed as

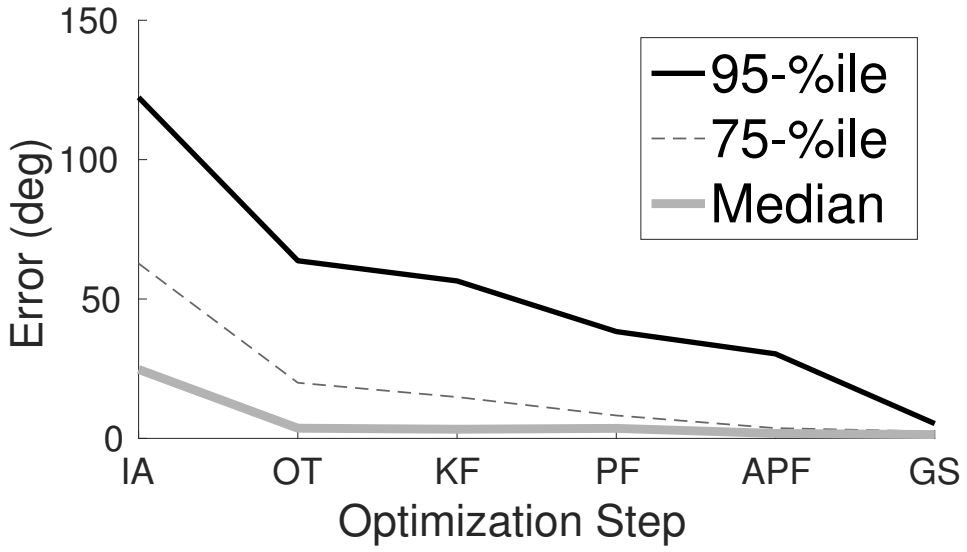

Figure 2.17: Errors vs Optimization Modules – Integer Ambiguity Resolution (IA), Outlier Elimination (OT), Kalman Filter (KF), Simple Particle Filter (PF), Adjusted Particle Filter (PF) and Multi-GNSS (GS) together contribute to the performance of SafetyNet

vision-based IMU supplements. Vision techniques are complementary to iMob. Offline, we leverage high-fidelity structure from motion to validate our accuracy.

2.8 Future Work

We briefly discuss a few points of future work.

• **Fusion with IMU**: SafetyNet is a completely orthogonal solution, but it is possible to combine GPS with IMU for even better accuracy. We leave this to future work.

• **GPS Sampling Rate**: Sampling rate of our GPS modules was only 5 Hz, but all of our techniques are fundamentally applicable to high rate GPS receivers.

• **Latency**: The processing power on drones continues to evolve – various drones offer diverse computing capabilities. Our future work would need to carefully profile the latency to charac-

terize the kinds of drones that could execute SafetyNet in real-time.

• **Vehicles**: Orientation estimation of self driving cars can also benefit from SafetyNet, adding an extra layer of reliability.

2.9 Conclusion

Grand visions abound for applications of drones. While Amazon, Google, and others have made great strides towards package delivery, etc., drones remain an unacceptable hazard to persons and property. By comprehensively addressing IMU failure through GNSS – a failover completely orthogonal to inertial methods, SafetyNet progresses the state of the art in drone safety.

Chapter 3

Bringing IoT to Sports Analytics

3.1 Introduction

Sports analytics is a thriving industry in which motion patterns of balls, racquets, and players are being analyzed for coaching, strategic insights, and predictions. The data for such analytics are sourced from expensive high-quality cameras installed in stadiums, processed at powerful backend servers and clouds. We explore the possibility of significantly lowering this cost barrier by embedding cheap Inertial Measurement Unit (IMU) sensors and ultrawide band (UWB) radios inside balls and players' shoes. If successful, real-time analytics should be possible anytime, anywhere. Aspiring players in local clubs could read out their own performance from their smartphone screens; school coaches could offer quantifiable feedback to their students.

Our work follows a growing excitement in IoT based sports analytics. Sensor-enabled football helmets, aimed at detecting concussions and head injuries, are already in the market. Nike is prototyping IMU-embedded shoes [86, 87], while multiple startups are pursuing ideas around camera-embedded jerseys [88], GPS-enabled soccer balls [89], and bluetooth frisbees [90]. However, we have not found a serious effort to accurately characterize 3D ball motion, such as trajectory, orientation, revolutions per second, etc.

The rich literature in wireless localization and inertial gesture recognition does not apply directly. WiFi-like localization infrastructure is mostly missing in the playground, and even if deployed, is not designed to support *cm-scale* 3D location at ball speeds. Inertial sensors such as accelerometers do not measure gravity when the ball is in free fall, since these sensors

detect only reactive forces. Worse, gyroscopes saturate at around 6 revolutions per second (rps) [91], while even an amateur player can spin the ball at $12rps$. In general, tracking a fast moving/spinning object in an open playground presents a relatively unexplored context, distinct from human-centric localization and gesture tracking applications.

In approaching this problem top-down, we develop multiple wireless and sensing modules, and engineer them into a unified solution. The technical core of our system relies on using ultrawide band (UWB) radios to compute the time of flight (ToF) and angle of arrival (AoA) of the signals from the ball. When this proves inadequate, we model the ball's physical motion as additional constraints to the underdetermined system of equations. Finally, we fuse all these sources of information into a non-linear error minimization framework and extract out the parameters of ball trajectory.

Spin estimation poses a different set of challenges. We need to determine the initial orientation of the ball at its release position and then track the 3D rotation through the rest of the flight. With unhelpful accelerometers and gyroscopes, we are left with magnetometers. While magnetometers do not capture all the dimensions of rotation, we recognize that the uncertainty in the ball's spin is somewhat limited since air-drag is the only source of torque. This manifests on the magnetometer as a sinusoidal signal, with a time varying bias (called "wobble"). We formulate this as a curve-fitting problem, and jointly resolve the ball's angular velocity as well as "wobble". In general, we learn that magnetometers can serve as gyroscopes in free-spinning objects.

Our experiment platform is composed of an Intel Curie board (IMU + UWB) embedded in the ball by a professional design company [92]. Two small UWB receiver boxes, called *anchors*, are also placed on the ground – additional anchors are infeasible due to the field layout in the Cricket game, discussed shortly. For ground truth, we use 8 Vicon based IR cameras positioned

at 4 corners of the ceiling. IR markers are pasted on the ball to enable precise tracking ($0.1mm$ and 0.2° for location and orientation). Since the ViCon coverage area is $10x10x4m^3$ – around half of the actual Cricket trajectories – we scale-down the length of the throws while maintaining realistic speed and spin.

Reported results from 100 different throws achieve median location accuracy of $8cm$ and orientation errors of 11.2°, respectively. A player (wearing a clip-on UWB board) is also tracked with a median error of $1.2m$ even when he is at the periphery of the field ($80m$ away from the anchor). All results are produced at sub-second latency, adequate for real time feedback to human players.

There is obviously room for continued research and improvement. First, we have sidestepped the energy question. In future, perhaps wireless charging will mitigate this problem; perhaps fast rotation will automatically scavenge energy. For now, our solution allows a battery life of ≈ 75 minutes between re-charges, permitting short training sessions. Second, our aerodynamic motion models are simplistic and did not get stress-tested in indoor settings– this may have yielded favorable results. Moreover, we could not exceed throw speeds beyond 45 miles/hour and 12 revolutions/s, both of which are around half of the professionals. Finally, this chapter focuses on Cricket, and although we believe our techniques are generalizable with modest modifications, we have not verified these claims. Our ongoing work is focussed on adapting *iBall* to baseball and frisbee.

To summarize, the contributions of this project are:

- *Formulating object tracking as an information fusion problem under the practical constraints of Cricket.* Designing an optimization framework for fusing time of flight (ToF) measurements, air-drag motion models, and noisy angle of arrival (AoA) estimates, to ultimately achieve desired accuracy.

- *Identifying various opportunities arising from free-fall motion.* Harnessing the magnetometer to jointly estimate rotation and rotation axis, thereby emulating an inertial gyroscope in free-fall scenarios.

The rest of the chapter expands on these technical components woven together by significant engineering effort. We begin with some background on Cricket, followed by challenges, opportunities, design, and implementation.

3.2 Background and Platform

3.2.1 Quick Primer on Cricket

We summarize the basic rules of cricket for those unfamiliar with the game. A Cricket match is divided into two sessions – in any session, one team is called the *batting side* and the other is called the *bowling* or *fielding side*. The teams switch roles in the second session. A playing *pitch* is located at the center of the field, with two *wickets* on each side of the pitch. A wicket is a set of 3 wooden sticks placed vertically one beside the other (see Fig.3.1). A player from the batting side stands in front of a *wicket* while a player from the bowling side runs up to the other wicket and throws the ball towards the batsman. All other players of the bowling side are called *fielders* and stand scattered around the park.

The bowler's objective is to hit the wicket with the ball, or to force the batsman to hit the ball in a way that a fielder can *catch* the ball before it drops to the ground. If the bowler is successful, the batsman is *out*, i.e., he goes off the field and the next batsman of the batting team comes to face the bowler. The batsman's goal, on the other hand, is to not get out, and to also hit the ball so that it goes past the fielders and reaches the periphery of the park, called a *boundary*. If the ball bounces on the ground at least once before it crosses the boundary, then the batting side scores 4 more points (called *runs*); if the ball goes over the boundary without any bounce,

6 runs are added to the team's score. A session ends when either N deliveries have been bowled or all the 11 batsmen are out, whichever occurs earlier. At the end , the team with a higher total score wins.

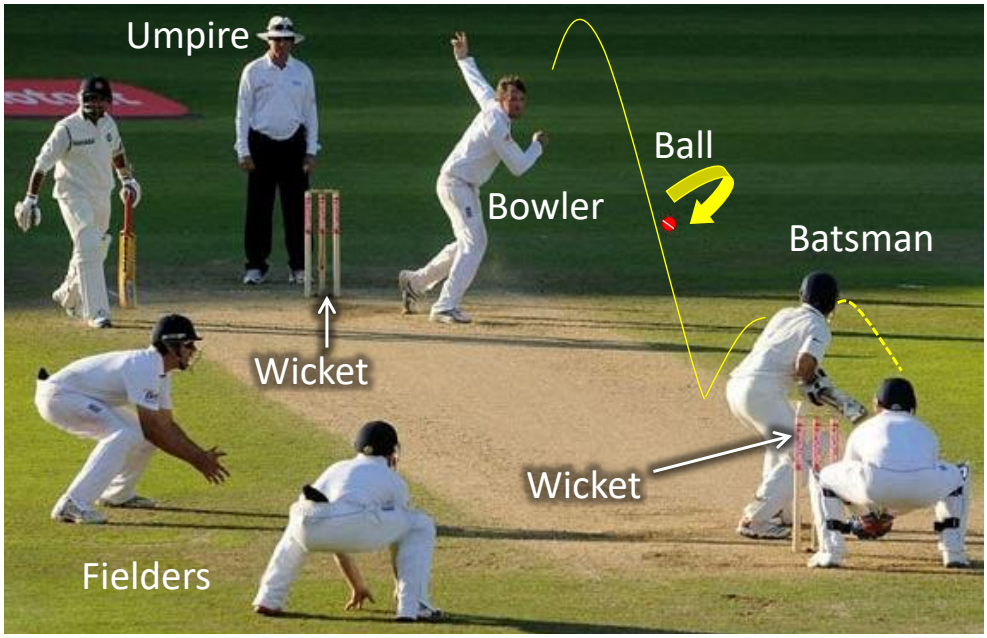

Figure 3.1: Cricket in action. Two sets of wickets placed at the bowler's and batsman's end.

Figure 3.2: Ball instrumentation: (a) Intel Curie board with IMU sensors and UWB radio. (b) Scooped out cricket ball for snug fit of the sensor box. (c) Closed ball with sensor box. (d) UWB 4-antenna MIMO radio serving as an anchor.

Analogy to Baseball: The similarity between Cricket and Baseball, from the perspective of ball and player tracking, is noteworthy. The baseball travels and spins at comparable speeds and rates, while the length of the "pitch" is also similar. Differences might arise from the stitching

patterns on the ball, and the viability of placing multiple anchors in baseball (in contrast to 2 in Cricket). In this sense, perhaps the ball's trajectory tracking problem becomes simpler in the case of baseball, but the spinning question still remains relevant.

3.2.2 The Solution Space

There are obviously many approaches to ball and player tracking – we briefly discuss our deliberations for selecting the IMU/UWB platform.

- High end cameras used today are expensive ($\$100,000+$) [93] because they need to be far away; we considered placing cheaper cameras at the wickets. The benefit is that the ball need not be instrumented. However, with no markers on the ball, spin tracking and de-blurring is challenging even with the best cameras. Cheap cameras at the wickets experience similar problems, get occasionally occluded by players, suffer in low light, and cannot track fielders scattered in the field. Experiments with iPhone cameras yielded poor results even with colored tapes on the ball.

- RFIDs on the ball (and readers placed at wickets) pose a far less price point ($2000). However, the rapidly spinning RFIDs exhibit continuous disconnections [94]. Further, cricket balls are continuously rubbed to maintain shine, crucial to the ball's swing and spin – pasting antennas on the surface is impractical.

- WiFi based tracking solutions are also impractical under the constraints of high speed and spin, cm-scale accuracy, and availability of a very few base stations on the 2D ground (which makes 3D tracking difficult due to *dilution of precision* (DoP) [95, 96]). Trials with laser rangers [97] and acoustic reflection techniques [98] also proved pointless. Given the small cross-sectional area of the ball, the reflections from them yielded high false positives.

- Our choice to embed electronics in the ball, although cumbersome, proved practical for accuracy and coverage in the field. In discussion with 3D printing and design companies, we gained

confidence that embedding should be feasible even under impact. Finally, UWB radios offer time of flight capabilities, a pre-requisite for extremely fast moving balls (> 80+ miles/hour). Our overall cost is estimated at $250.

3.2.3 Instrumenting Balls and Anchors

Fig.3.2 illustrates the steps in ball instrumentation. A Quark CPU, IMU BMM150 sensors, and a Decawave UWB radio are cased in a plastic polymer box and snug-fitted into a hole (to avoid rattling). The two halves of the ball are closed shut and a hole drilled to bring out a USB port to the surface for recharging. The sensor data is stored on a local flash or can be streamed through the UWB radio to the nearby "anchor".

The anchor is a UWB receiver box placed at each wicket. The UWB radio from Decawave [99] is 802.15.4 compliant with support for 3.5 to 6.5 GHz bands (12 channels) and a bandwidth of 500 MHz (data rates of up to 6.8 Mbps). The radio operates under low power, with sleep current at 100 nanoAmp. While the ball contains a single antenna (due to space restrictions), a 4 antenna MIMO radio is fitted in the anchor (Fig.3.2(c)).

Fig.3.3 illustrates the overall deployment in real settings. UWB signals are exchanged between the ball and anchors to compute the ball's range as well as the angle of arrival (AoA) from the phase differences at different antennas. The range and AoA information are combined for trajectory estimation. For spin analytics, the sensors inside the ball send out the data for off-ball processing. Players in the field can optionally wear the same IMU/UWB device (as in the ball) for 2D localization and tracking.

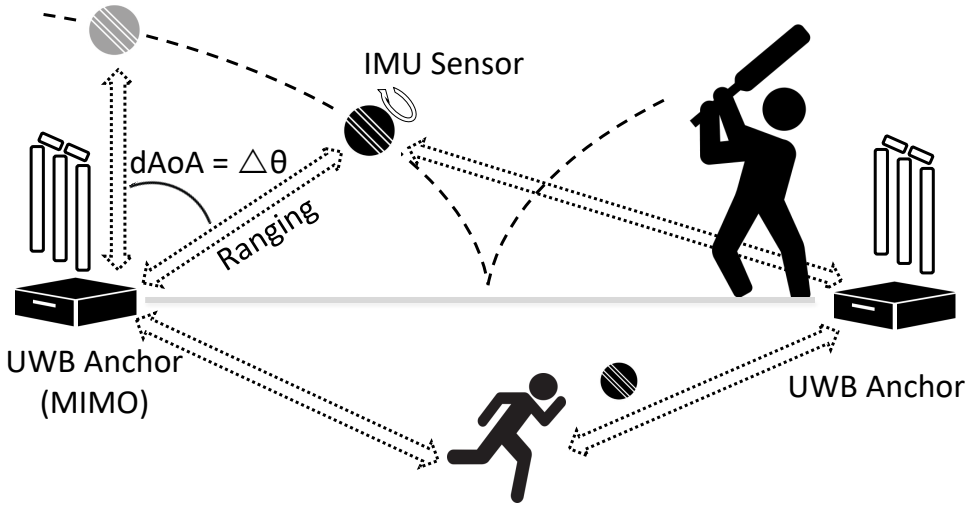

Figure 3.3: Two anchors and a ball deployed on the ground, while players optionally have the device in their shoes.

3.3 System Design: 3D Spin Tracking

Three main spin-related metrics are of interest to Cricketers: (1) revolutions per second, (2) rotation axis, and (3) seam plane[1]. From a sensing perspective, all these 3 metrics can be derived if the ball's 3D orientation can be tracked over time. This motivates a discussion on orientation, rotation, and coordinate frameworks.

3.3.1 Foundations of Orientation

The *orientation* of an object is the representation of the object's local X, Y, Z axes as vectors in the global coordinate frame. A *rotation* of an object is a change of orientation, and can be decomposed into sequence of rotations around its (local) X, Y, and Z axes. Put differently, any new orientation can be achieved by rotating the object (by appropriate amounts) on each of the 3 axes, one after the other. The gyroscope measures each of these rotations per unit time, called *angular velocity*. Thus, theoretically, if one knows the *initial orientation* of an object in the global coordinate frame, then subsequent orientations can be tracked by integrating the

[1]The seam is a stitch along the equator of the Cricket ball.

gyroscope–measured angular velocity across time.

Expressing the object's initial orientation in the global framework should be possible since gravity and magnetic North are both along globally known directions. Thus, the object's local axes can be rotated until the local representation of gravity and North align with the known global directions. We consider an example below.

Fig.3.4(a) shows a global frame $\{X_g, Y_g, Z_g\}$ with its X_g pointing East, Y_g pointing North, and Z_g pointing up against gravity. Fig.3.4(b) shows an object in an unknown orientation. Now, the object can be rotated around X axis until the measured gravity is along its own $-Z$ direction; it can be rotated again around this $-Z$ axis until the measured magnetic field (compass) is along its own Y. Now the local and global frameworks have fully aligned, and we denote the total rotation as a single matrix R:

$$\begin{bmatrix} X & Y & Z \end{bmatrix} R = \begin{bmatrix} X_g & Y_g & Z_g \end{bmatrix} \tag{3.1}$$

We define an object's orientation, R_O, as the inverse of this rotation matrix, R^{-1}. Intuitively, if an object needs a clockwise rotation of 30° to align with the global framework, then its orientation must be 30° counter-clockwise.

Thus, we have the capability to compute both initial orientation and angular velocity; from these, any spin-related analytics should ideally be trackable.

■ **Challenges with In-Flight Balls:** Challenges emerge in the real world and particularly in this cricket setting: (1) The gyroscope is noisy and this error accumulates since rotation is a time–integral of angular velocity (2) Worse, the gyroscope saturates beyond 5 revolutions/sec. (rps), while even amateurs can spin the ball at $12rps$ (professionals attain > 30). A full analysis of gyroscopes is outside the scope of this project. Gyroscope saturation behavior is rooted in im-

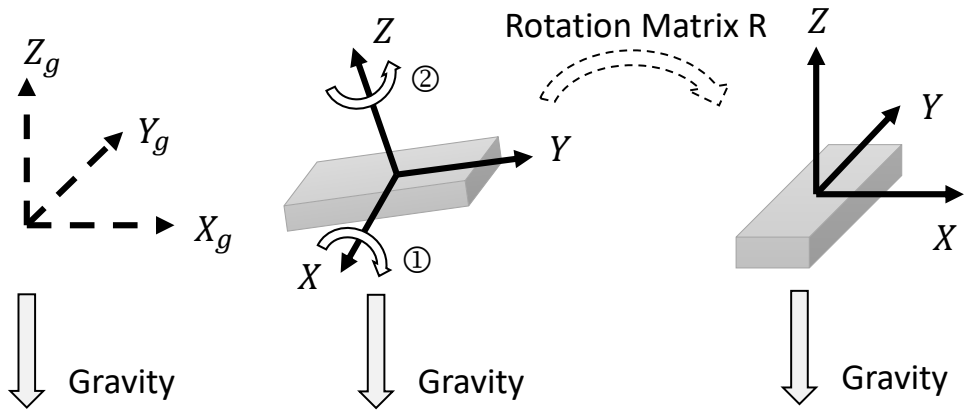

Figure 3.4: Rotating local axes to align local directions of gravity and magnetic North with the global directions.

perfections accumulated during the manufacturing process of MEMS sensors. These imperfections make gyroscope to exhibit nonlinear response to input vibrations. This translates to a super linear increase in error with increasing angular velocities [100, 101]. On the other hand, attempts to reduce imperfections increases the manufacturing cost or decreases the yield rate.

(3) An accelerometer under rest measures the direction of gravity. This information can be used to align the local and global coordinate frames as discussed earlier in this section. However, gravity is not measured in accelerometers during free-fall flights which precludes opportunities to rotate and align the local coordinate frame.

In sum, known techniques cannot compute initial orientation or rotation when the ball is in flight.

3.3.2 The Core Opportunity

At a high level, 2 observations are central.

- In the absence of air-flow, there is no external torque on the ball, implying that the ball's rotation is restricted to a single axis throughout the flight (i.e., the axis around which it was rotated by the bowler).

- From the ball's local reference frame, the magnetic North vector spins around some axis. Given

a single rotation axis, the magnetometer can indeed infer the axis and measure both magnitude and direction of rotation.

Of course, air-drag pollutes this opportunity since the ball begins to experience additional rotations. This poses the main challenge. An illustrative example follows.

3.3.3 An Illustrative Example

Let's assume the ball's mass is symmetrically distributed and its center of mass is precisely at the center. Let's also consider gravity forces alone and no air drag. Now, due to conservation of angular momentum, the ball will not change its rotating state because no torque is generated from gravity. The dimension of this rotational motion is limited to 1 since the motion can be continuously expressed around a single axis, R^G. Fig.3.5 illustrates the situation – each local X, Y, Z axis rotates in different cones around the same R^G. As an aside, the magnetic North is also a fixed vector N^G in the global framework (henceforth, we use superscript G/L to indicate that a vector is being observed in the global/local framework).

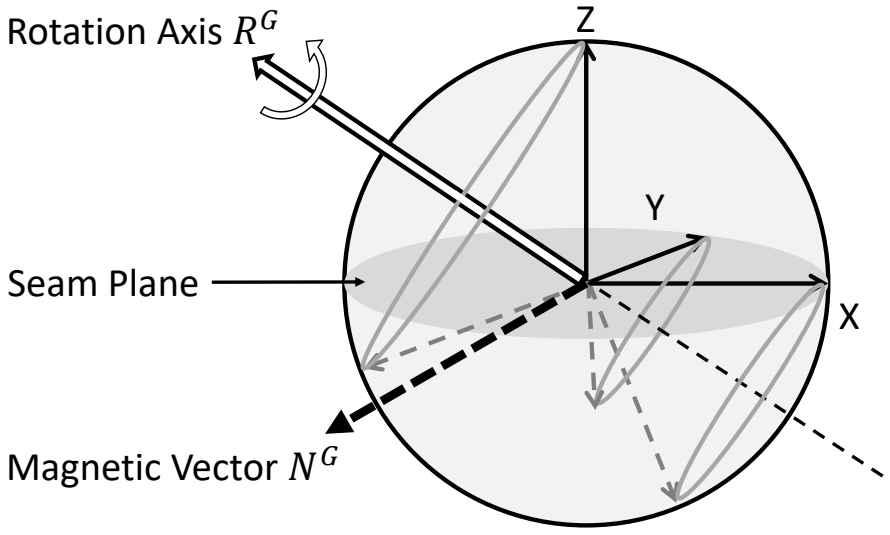

Figure 3.5: In the global framework, the ball in rotating around a constant rotation axis R^G.

Shifting our perspective from the global to local coordinate system, Fig.3.6(a) shows that the local X, Y, and Z axes are now fixed, but the magnetic vector N^L rotates in a cone around a fixed

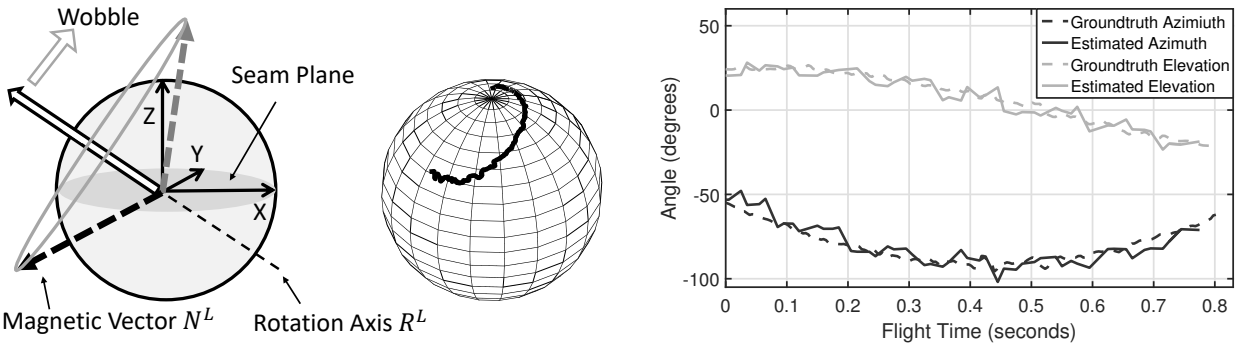

Figure 3.6: (a) In the local framework, the magnetic vector N^L is rotating around local rotation axis R^L. (b) In the local framework, local rotation axis R^L is slowly moving. (c) One example of $R^L_{(t)}$ estimation from cone fitting

local vector R^L. Since magnetometers can reliably measure a single dimension of rotation, it should be possible to measure the parameters of this cone. This is the core opportunity – the low-dimensional mobility during free-fall empowers the magnetometer to serve as a gyroscope.

Unfortunately, with air-drag, the ball still continues to rotate around the same global axis R^G, but experiences an additional rotation along a changing axis. To envision this, consider the ball spinning around the global vertical axis with the seam on the horizontal plane. With air-drag, the ball can continue to spin around the identical vertical axis, but the seam plane can gradually change to lie on the vertical plane. This is called "wobble" and can be modeled as a varying local rotation axis, R^L. Fig.3.6(b) shows the locus of R^L as it moves in the local framework (this was derived from ViCon ground truth). Thus, the center of the N^L cone is moving on the sphere surface, even though the width of the cone remains unchanged. This derails the ability to compute rotations from magnetometers.

3.3.4 Problem Formulation

Based on the earlier discussion, we know that if two non-collinear vectors can be observed in the local framework, and their representations known in the global frame, then the orientation

of the object can be resolved. We mathematically express the orientation of the ball at time t as a rotation matrix, $R_{O(t)}$. This matrix is a function of the globally fixed vectors (i.e., rotation axis and magnetic North) and their locally measured counterparts.

$$R_{O(t)} = \begin{bmatrix} R^G & N^G & R^G \times N^G \end{bmatrix} \begin{bmatrix} R^L_{(t)} & N^L_{(t)} & R^L_{(t)} \times N^L_{(t)} \end{bmatrix}^{-1} \tag{3.2}$$

Here R^G and N^G are the rotation axis and magnetic North vectors, respectively – both are in the global framework and are constant during flight. The third column vector, $(R^G \times N^G)$, is a cross product necessary to equalize the matrix dimensions on both sides. $N^L_{(t)}$ is the local magnetic vector measured by the magnetometer, $\begin{bmatrix} m_x & m_y & m_z \end{bmatrix}^T$. $R^L_{(t)}$ is the local rotation axis which is slowly changing during the flight of the ball. From previous discussion we know that $R^L_{(t)}$ is always the centerline of the instantaneous $N^L_{(t)}$ cone.

Our goal now is to estimate two of the unknowns, namely R^G and time varying $R^L_{(t)}$. We know that R^G remains constant hence resolving it at the beginning of the flight will suffice – the same value can be used all the way till the end. For $R^L_{(t)}$, we know that it is moving on the sphere of the ball and the magnetic North is constantly rotating around it. We focus on tracking $R^L_{(t)}$ first and then address R^G.

3.3.5 Tracking Local Rotation Axis $R^L_{(t)}$

Since $N^L_{(t)}$ forms a cone around $R^L_{(t)}$, tracking $R^L_{(t)}$ is equivalent to tracking the centerline of the cone. Now, given that 3 non-coplanar unit vectors determine a cone, a straightforward idea is to fit a cone using 3 consecutive measurements: $N^L_{(t-1)}$, $N^L_{(t)}$ and $N^L_{(t+1)}$. Fig.3.6(c) shows the result: the estimation follows the true $R^L_{(t)}$ trend, but is considerably noisy.

These noise in $R^L_{(t)}$ estimation will translate to orientation error according to Equation 3.2. The noise cannot be reduced by fitting the cone over larger number of magnetometer measure-

ments – this is because the cone would have moved considerably within a few sampling intervals. Our observation is that, because the ball's flight time is short (less than a second), we can effectively describe the (azimuth and elevation[2]) changes in $R^L_{(t)}$ as a quadratic function of time t. Formally:

$$R^L_{(t)} = \begin{bmatrix} \cos(\theta_t)\cos(\varphi_t) \\ \cos(\theta_t)\sin(\varphi_t) \\ \sin(\theta_t) \end{bmatrix} \tag{3.3}$$

$$\text{Elevation } \theta_t = A_{el}\, t^2 + B_{el}\, t + C_{el} \tag{3.4}$$

$$\text{Azimuth } \varphi_t = A_{az}\, t^2 + B_{az}\, t + C_{az} \tag{3.5}$$

Put differently, we model the motion of a moving cone, under the constraints that the center of cone is moving on quadratic path (on the surface of the sphere) and that the cone angle $\theta_{NR} = \angle\left(N^L_{(t)}, R^L_{(t)}\right)$ is constant. We pick the best 6 parameters of this model that minimize the *variance* of the cone angles as measured over time. Our objective function is:

$$\underset{6\,\text{paras}}{\text{argmin}}\ \text{Var}\left[\angle\left(R^L_{(T)}, N^L_{(T)}\right), \angle\left(R^L_{(T+1)}, N^L_{(T+1)}\right), \cdots\right] \tag{3.6}$$

where T is the moment the ball is released. The initial condition to this optimization function is derived from a smoothened version of the basic cone fitting approach, described in Fig.3.6(c).

3.3.6 Track Global Rotation Axis R^G

Fig.3.7 shows 2 phases of ball tracking: *pre-flight* and *in-flight*. The above section described phase 2, the tracking of $R^L_{(t)}$ when the ball is spinning *in-flight*. However, recall that the global rotation axis, R^G, also needs to be estimated to solve for orientation $R_{O(t)}$ in Equation 3.2.

Tracking R^G during the ball's flight is difficult. Sensor data during the flight only tells us where

[2]Azimuth and elevation are latitudinal and longitudinal directions on a sphere's surface: a point on the sphere can be expressed as a tuple of these 2 angles.

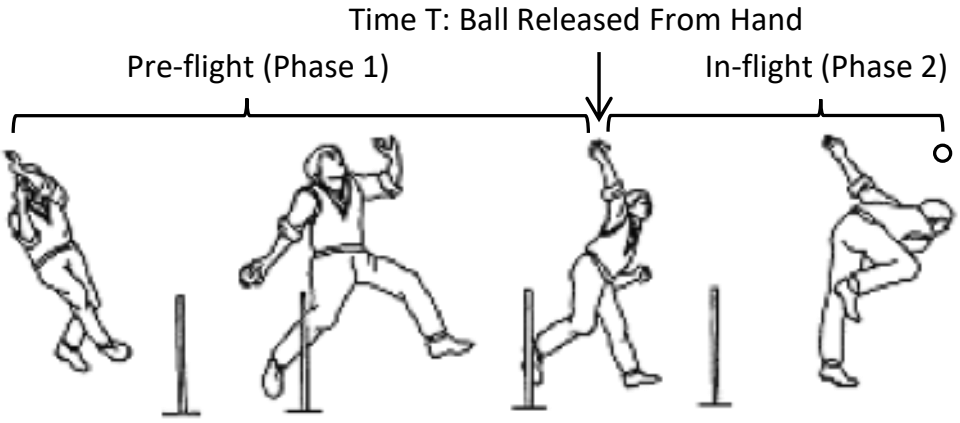

Figure 3.7: Two phases of ball motion.

$R^L_{(t)}$ is pointing (center of the $N^L_{(t)}$ cone) but it does not reveal any information about R^G. Fortunately, the rotation axis R^G and magnetic vector N^G are two constant vectors. The angle between these two vectors, $\angle\left(N^G, R^G\right)$ is the same as the local $N^L_{(t)}$ cone angle θ_{NR}. Thus, R^G can only lie on a cone around N^G whose cone angle is θ_{NR}. Although useful, it's insufficient – we still do not know the point on the cone's circle corresponding to R^G.

To mitigate this problem, we focus on sensor measurements in phase 1 (pre-flight). Since this is not free-fall, and the ball is not spinning fast, the gyroscope and accelerometer are both useful. Our aim is to identify a stationary time point to compute the initial orientation of the ball, and use the gyroscope thereafter to integrate rotation until the point of release, T. Once we obtain orientation at T, denoted $R_{O(T)}$, we simply use the following equation to solve for the global rotation axis $R^G_{(T)}$

$$R^G_{(T)} = R_{O(T)}\, R^L_{(T)} \tag{3.7}$$

Then, we use $R^G_{(T)}$ as our estimation of R^G for the whole flight in Phase 2.

In general, gyroscope noise and saturation can render $R^G_{(T)}$ erroneous. However, since the ball does not spin while in the hand (in fact, it rotates less than 1 revolution), and the angular velocity saturates the gyroscope only at the last few moments before ball-release, we calibrate $R^G_{(T)}$

using the cone angle restriction mentioned above. Fig.3.8 reports consistently small $R^G_{(T)}$ error from 50 experiments.

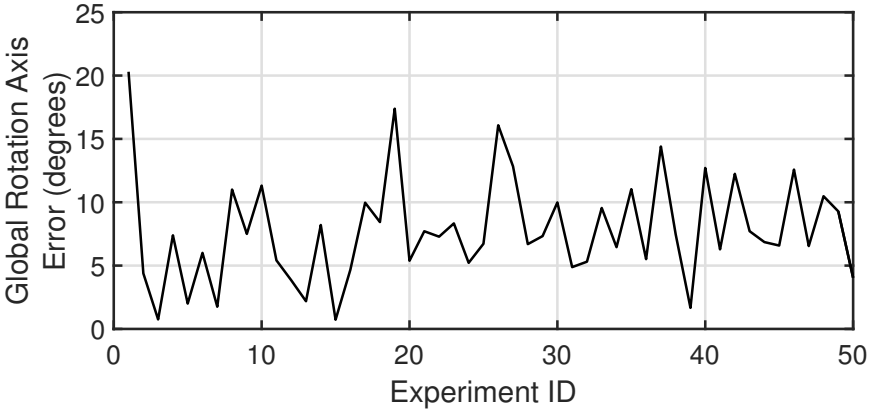

Figure 3.8: Error in estimating global rotation axis $R^G_{(T)}$ is reasonably small across 50 experiments.

In conclusion, gyroscope dead-reckoning right before ball release, combined with local rotation axis tracking during flight, together yields the time-varying orientation of the ball. Algorithm 3.1 presents the pseudo code for final overall solution.

Algorithm 3.1: Ball Orientation Tracking During Flight

1: Get coarse $R^L_{(t)}$ by combining 3 consecutive magnetometer measurements
2: Use them as the initial starting point to search for parameters that minimize Var $\left[N^L_{(t)}$ cone angles$\right]$
3: Compute cone angle θ_{RN} = Mean $\left[N^L_{(t)}$ cone angles$\right]$
4: Use gyroscope to tracking ball's orientation at the release time, $R_{O(T)}$
5: Get global rotation axis during flight: $R^G = R_{O(T)}\, R^L_{(T)}$
6: Calibrate R^G using θ_{RN}.
7: Use Equation 3.2 to compute ball's orientation at any time t during flight

3.4 System Design: 3D Trajectory Tracking

Location related analytics are also of interest in Cricket. 3 main metrics are: (1) distance to first bounce, called *length*, (2) direction of ball motion, called *line*, and (3) *speed* of the ball at the end of the flight. These metrics are all derivatives of the ball's *3D trajectory*. Our approach to

estimating 3D trajectory relies on formulating a parametric model of the trajectory, as a fusion of the *time of flight* (ToF) of UWB signals, *angle of arrival* (AoA), physics motion models, and DoP constraints (explained later). A gradient decent approach minimizes a non-linear error function, resulting in an estimate of the trajectory. We present technical details next.

3.4.1 Ranging with UWB

The Decawave UWB radios offer time resolution at 15.65ns. With modest engineering, we were able to compute the ToF and translate it to range measurements (with 15cm error). Briefly, the ball sends a POLL, the anchor sends back a RESPONSE, and the ball responds with a FINAL packet. Using the two round trip times, and the corresponding turn-around delays, the time of flight is computed without requiring time synchronization between the devices (algorithm details in [99, 102]). Multiplied by the speed of light, this yields the ball's range. This is not our contribution since the Decawave platform offers the foundational capabilities.

Observe that UWB ranging is available from only 2 anchors (placed at the two wickets) and therefore inadequate to resolve the 3D location of the ball. Additional anchors cannot be placed on the ground since it will interfere with the motion of the ball and fielders, while placing anchors outside the field (90m away from the wickets) significantly degrades SNR and ranging accuracy. Fig.3.9 shows the intersections of the 2 anchor measurements, i.e., circles formed by the intersection of two spheres centered at the anchors. At a given time, the ball can lie on any point of a circle. Given that the initial position and velocity of the ball is unknown, many 3D trajectories can satisfy these constraints.

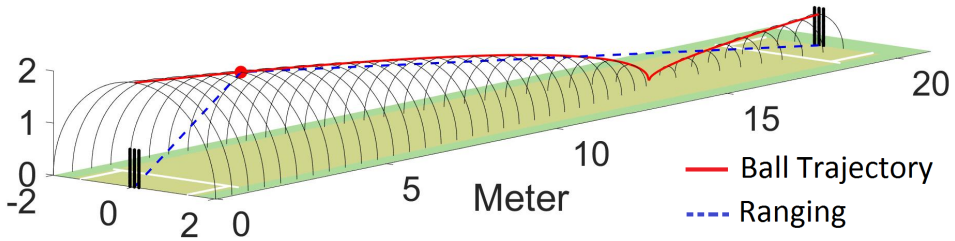

Figure 3.9: Intersections of ranging measurements leave one location dimension unresolved.

3.4.2 Mobility Constraints

We bring two mobility constraints to resolve the uncertainty: (1) physics of ball motion, and (2) opportunities from the ball's bouncing position.

(1) Physics of Ball Motion

Fig.3.10 shows a free-body diagram depicting the forces acting on the Cricket ball while in flight. Besides gravity, aerodynamic forces are acting on the ball. Briefly, the ball surface is smooth on one side of the seam and rough on the other (Cricket bowlers continue to polish the smooth side during the game). This disparity causes unbalanced air-flow, causing a side force. The speed of the ball can cause a slight air drag force. The magnitude and direction of the side forces depends on the seam orientation, surface roughness, and ball velocity. The drag and side force coefficients can be approximated as constants [103]. The side force can produce up to $1m$ of lateral deflection in trajectory.

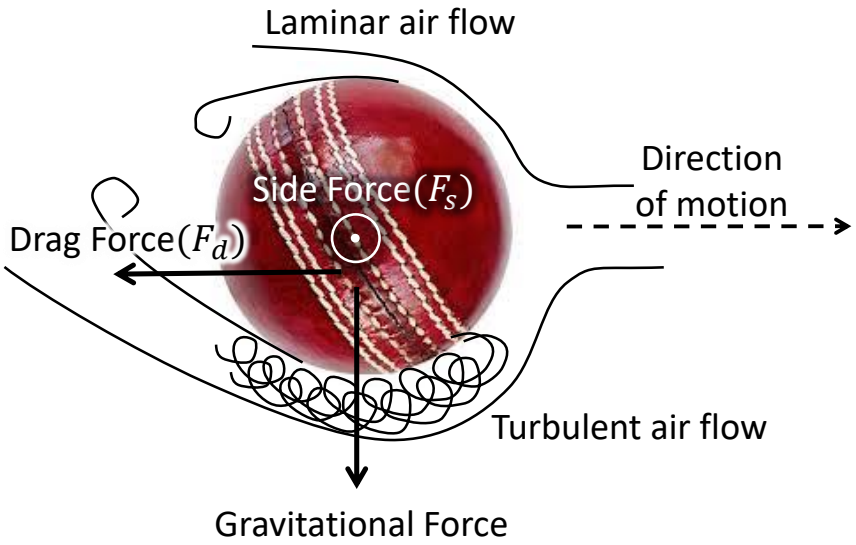

Figure 3.10: Unbalanced air-flow due to asymmetric smoothness on the ball's surface causes side force.

Under the above forces, ball motion follows a simple projectile path [103].

Fig.3.11(a) shows the extent to which the projectile model (without the aerodynamic forces)

fits the ball's true trajectory (derived from ViCon). The projectile was seeded by the ViCon-computed initial location and initial velocity. The median error is $1cm$ across 25 different throws of the ball, offering confidence on the usability of the model in indoor environments. The efficacy outdoors remains an open question.

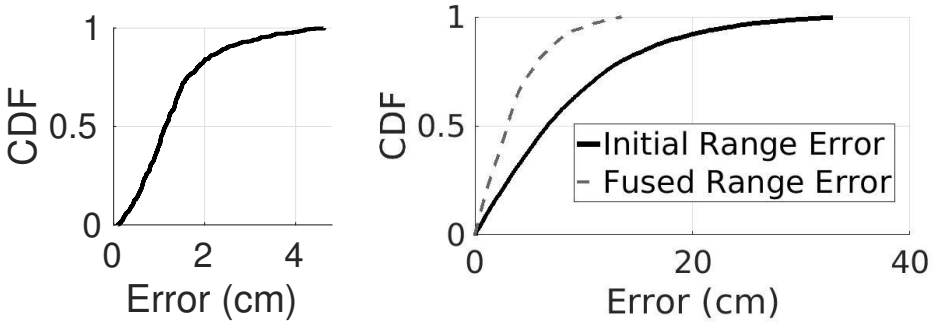

Figure 3.11: (a) Error between a parametric motion model and ground truth (derived from ViCon cameras). (b) Reduced ranging error after fusing UWB ranging with parameterized motion models.

(2) Bouncing Constraint

When the ball bounces before reaching the batsman (detectible from an accelerometer spike), the Z component of location – the ball's height – is 0. This resolves the uncertainty at a single point, i.e., in combination with UWB ranging, the ball's location can be computed *only* at this point. Thus, one point on the trajectory is "pinned down", shortlisting a smaller set of candidate trajectories. We can now fuse these physical constraints with the underdetermined system from Fig.3.9.

On the other hand, there might be cases where the ball does not bounce. One possibility is to work with the constraints of physics of ball motion alone which comes at a cost of slight degradation in accuracy. We are considering an alternative possibility of imposing a stronger constraint based on the location of impact of the ball on the bat. By embedding wearable sensors on the bat/batsman, we believe this is feasible and leave further exploration to future work.

3.4.3 Fusing Range and Motion Constraints

Our goal is to model the trajectory as an error minimization problem. We denote the two anchor positions as $(x_{ia}, y_{ia}, z_{ia})\ \forall\ i \in \{1,2\}$. Also, we denote the initial location and initial velocity of the ball – at the point of release from the hand – as (x_o, y_o, z_o) and (v_x, v_y, v_z), respectively. Thus, at a given time t, the estimated location of the ball from simple physics models (without aerodynamics) is:

$$S_{xe}(t) = x_o + v_x t \tag{3.8}$$

$$S_{ye}(t) = y_o + v_y t \tag{3.9}$$

$$S_{ze}(t) = z_o + v_z t - 0.5 g t^2 \tag{3.10}$$

Here, g is the acceleration due to gravity. Using this, the range from each anchor i is parameterized as:

$$R_{i,p}(t) = \sqrt{(S_{xe} - x_{ia})^2 + (S_{ye}(t) - y_{ia})^2 + (S_{ze}(t) - z_{ia})^2} \tag{3.11}$$

Once we have the range modeled, we design the error function, Err, as a difference of the parameter-modeled range and the measured range as follows.

$$\underset{6params}{\operatorname{argmin}}\ Err = \sum_{i=1,2} \sum_{t} \{R_{i,p}(t) - R_{i,m}(t)\}^2 \tag{3.12}$$

This objective function is minimized using a gradient descent algorithm, however, since it is highly *non-convex*, multiple local maxima exist. We bound the search space based on 2 boundary conditions: (1) The Z coordinate of the bouncing location is zero. (2) The initial ball-release location is assumed to be within a $60cm^3$ cube, as a function bowler's height.

While this proved effective in eliminating many local maxima, Fig.3.11(b) shows that the median ranging error is $3cm$. However, translating range to location is affected by a phenomenon called *dilution of precision* (DoP) [95].

3.4.4 Dilution of Precision (DoP)

Ideally, the intersection of two UWB range measurements (i.e., two spheres centered at the anchors) is a circle – the ball should be at some point on this circle. In reality, ranging error causes the intersection of spheres to become 3D "tubes". Now, when the two spheres become nearly tangential to each other, say when the ball is near the middle of two anchors, the region of intersections becomes large. Fig.3.12(b) shows this effect. This is called DoP and seriously affects the location estimate of the ball (later we will see how DoP in Fig.3.12(c) affects the localization of the players). DoP is a fundamental problem that affects other trilateration applications like GPS.

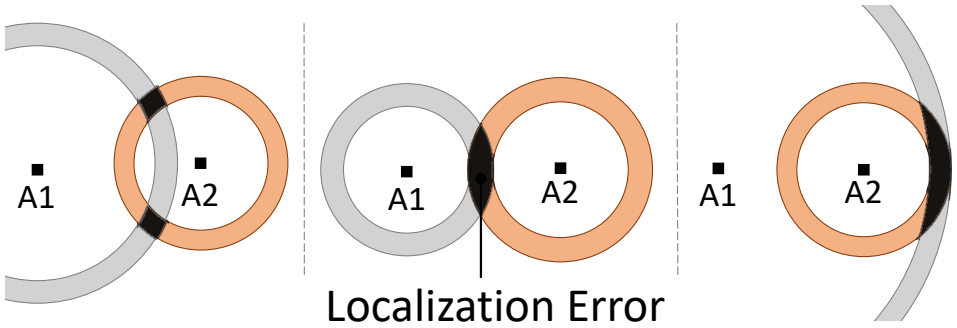

Figure 3.12: DoP introduced from ranging errors: (a) Lower DoP when ranging circles not tangential, (b) higher DoP when circles externally tangential, (c) max DoP when circles internally tangential.

Fig.3.13 shows the error variation as the ball moves in flight. The ball is released at time $t = 0$ and it reaches the batsman by the end of the flight. – clearly, the error increases and is maximal near the middle of the flight. However, since the DoP can be modeled as a function of distance from the anchors, it should be possible to *weigh* the errors in the minimization function as

follows:

$$\underset{6params}{\operatorname{argmin}} \; Err = \sum_{i=1,2} \sum_{t} \{(R_{i,p}(t) - R_{i,m}(t)) \times \frac{1}{\sqrt{(DoP)}}\}^2 \tag{3.13}$$

This revised minimization function pays less importance to range measurements weighted by a large DoP. Results improve to a median of $16cm$ error.

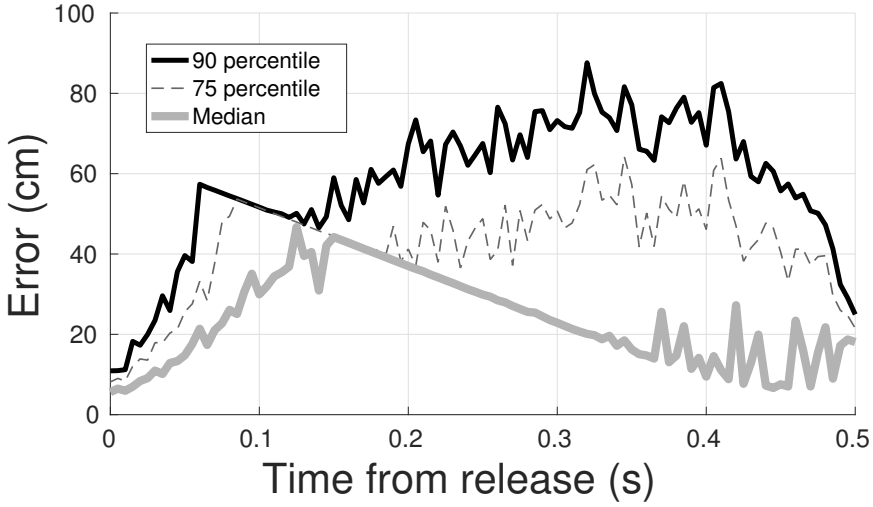

Figure 3.13: DoP aggravates error near the middle.

3.4.5 Exploiting Angle of Arrival (AoA)

The MIMO antennas at the anchors are capable of synchronized phase measurements of the incoming signal. Fig.3.14 shows how the phase difference ϕ is a function of the difference in signal path (p_1 and p_2), which is in turn related to AoA, θ. Thus, we have:

$$d\cos(\theta)\frac{2\pi}{\lambda} = \phi \tag{3.14}$$

$$\cos(AoA) = \cos(\theta) = \frac{\phi\lambda}{2\pi d} \tag{3.15}$$

We employ a MIMO receiver only on the bowler side (the other anchor cannot be utilized since it gets significantly interfered by the batsman, corrupting phase measurements). Now, for this

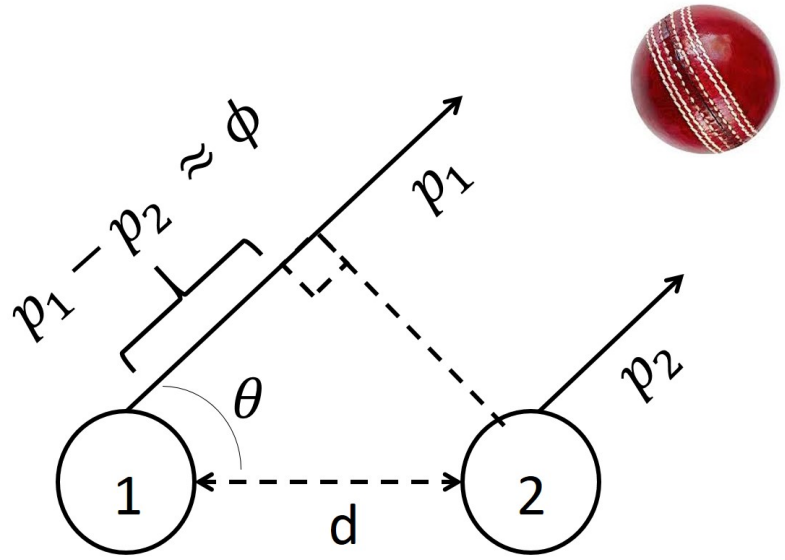

Figure 3.14: AoA is derived from phase differences.

single anchor, say the antennas are separated along the x-axis; then the AoA can be expressed in terms of ball location, anchor locations, and measured ranges:

$$\cos(\theta) = \frac{S_x - x_{ia}}{R_{ia}} \tag{3.16}$$

$$S_{x,aoa} = \cos(\theta) R_{ia} + x_{ia} \tag{3.17}$$

Thus, it is possible to refine the previous estimates of the trajectory by including AoA in the error function. Finally, DoP problems arise with AoA too – as the ball travels further away from the anchor, the location error increases for a small $\Delta\theta$ error in AoA. In fact, the error is $R\Delta\theta$, where R is the ball's range.

3.4.6 Exploiting Antenna Separation

It is clear from Equation 3.15 that the AoA error is a function of antenna separation d – higher antenna separation will decrease the error in measurement of $\cos\theta$ (AoA). However, with antenna separation higher than wavelength λ, the phase wraps and introduces ambiguity in

AoA estimation – called *integer ambiguity*. For unambiguous AoA measurements, $d\cos\theta < \frac{\lambda}{2}$ or $d < \frac{\lambda}{2}$. Fig.3.15(a) shows a common case of unambiguous AoA measurement during a ball throw. Evidently, AoA is heavily corrupted from spinning antenna orientation and polarization.

To mitigate the noise, we increase the antenna separation d. However, when $d > \frac{\lambda}{2}$, the ambiguous AoA measurements are indicated in the equation below.

$$d\cos(\theta) = \frac{\phi\lambda}{2\pi} + N\lambda \tag{3.18}$$

$$\cos(\theta) = \frac{\phi\lambda}{2\pi d} + N\frac{\lambda}{d} \tag{3.19}$$

The AoA is not only a function of the phase difference ϕ, but also a function of the unknown integer ambiguity N. Fortunately, the smooth trajectory of the ball provides an opportunity for tracking the integer ambiguity across measurements, thereby any wrap around can be detected and accounted for. Fig.3.15(b) shows a common case of AoA measurement (known integer ambiguity) for a ball throw after increasing the antenna separation to 18 cm – 2.5 times the wavelength. Evidently, the noise is much lower, offering an additional opportunity.

3.4.7 Fusion of AoA with Ranging/Physics

In order to fuse, AoA, we need to firstly resolve the integer ambiguity. Our technique to resolve this is simple. At any point during the gradient search algorithm, we obtain an estimated AoA from current set of parameters. We simply resolve the integer ambiguity by substituting the currently estimated AoA in Equation 3.19. Incorrect ambiguity resolution would automatically explode the error function because of mismatch with range measurements (For example, if the integer ambiguity resolution is incorrect even by a single integer, that would introduce a median mismatch of 7.5cm (one wavelength) between inferred and measured ranges). With integer ambiguity resolution, we are ready to update the objective function Err, with AoA fusion.

$$\underset{6params}{\operatorname{argmin}} \; Err = \sum_{i=1,2} \sum_{t} \{(R_{i,p}(t) - R_{i,m}(t)) \times \frac{1}{\sqrt{(DoP)}}\}^2 + \sum_{t} \{\frac{(S_{x,p}(t) - S_{x,aoa}(t))}{R_{aoa,p}\Delta\theta}\}^2 \quad (3.20)$$

where $S_{x,aoa}(t)$ is drawn from Equation 3.16. The $R_{aoa,p}\Delta\theta$ (R_{aoa} denotes range from AoA anchor, $\Delta\theta$ is AoA noise) factor decreases the weight for AoA measurements taken far away from the AoA Anchor.

To summarize, iBall incorporates noisy ranging and AoA measurements from UWB Anchors with physics based motion model to track the ball trajectory. Results are presented in Section.5.4.

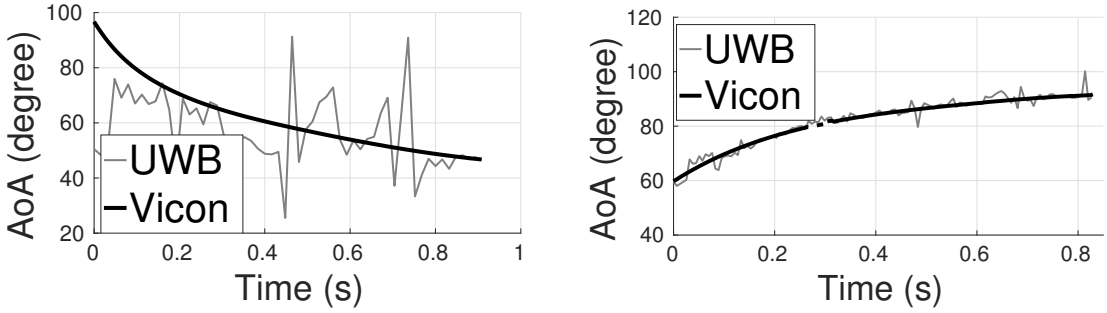

Figure 3.15: Improved AoA with antenna separation.

3.5 System Design: Player Tracking

iBall aims to track the movement of players in the field. Assuming clip-on UWB devices on the players, the ball ranging techniques should apply directly; in fact, since players are on the 2D ground, the tracking should be feasible with 2-anchor ranging alone. However, a different form of DoP emerges: when the two lines joining the player and the two anchors tend to get collinear, the ranging rings around the anchors begin to exhibit larger overlapping areas (see Fig.3.12(c)).

Fig.3.16 shows simulations of DoP on a real-sized Cricket ground. As the player moves closer to the X axis (i.e., higher collinearity), the 90 percentile uncertainty of estimated location increases to $15m$, in contrast to $1m$ when the player is perpendicular to the anchors. The effect is worse with higher distance from anchors.

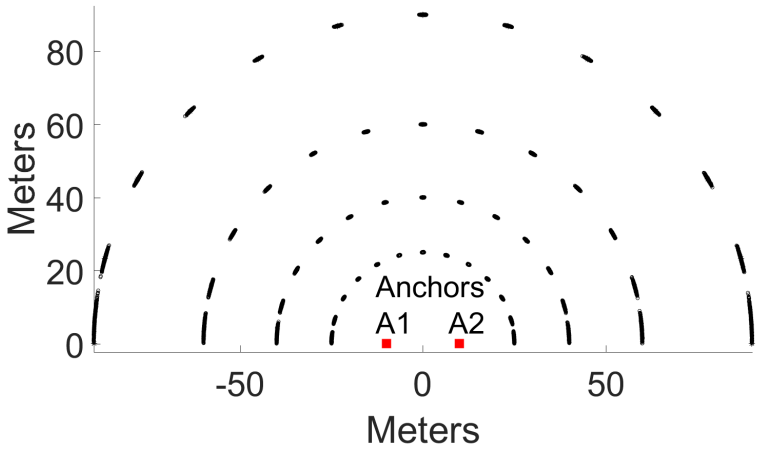

Figure 3.16: Simulation of estimated player location.

3.5.1 DoP Suppression through Filtering

To cope with DoP degradations, we apply Kalman Filtering (KF) to player tracking. The basic idea is to detect (from the accelerometer) that a player has started running, and combine the motion model of a human-run with the UWB ranging estimates. For the human-run model, we assume the velocity to be piecewise constant in short time scales (second). The velocity is periodically updated using the recent KF estimates, thereby accounting for changes in the human's run patterns. Section 5.4 will show results from real experiments on a large field. iBall applies the same techniques to the ball, which can also be tracked after it has been hit by the batsman. We evaluate the overall system next.

3.6 Evaluation

Our experiments were performed in a $10x10x4m^3$ indoor space with 8 ViCon IR cameras installed on the ceiling for ground truth (Fig.3.17(a)). Fig.3.17(b) shows IR markers pasted on the ball – this enables location and orientation tracking accuracies of $0.1mm$ and 0.2°, respectively. The authors pretended to be Cricket players and threw the ball at various speeds and spins (for a total of 100 throws). A batsman was realistically positioned to create signal blockage between the ball and the anchor. The Intel curie chip provides IMU data at $70Hz$, while the anchors perform ranging/AoA at $150Hz$.

Metrics: At any given time t during the flight of the ball, ViCon camera provides the true orientation of the ball, say $C(t)$, while iBall generates an estimated orientation, say $E(t)$. The **Orientation Error (ORE)** is essentially the minimum rotation that must be applied to $E(t)$ to align with $C(t)$. We measure this error across different values of t and plot the CDF. We also plot the angle difference between the rotation axis and the seam plane, called **Rotation Axis Error (AXE)** and **Seam Plane Error (PLE)**. Now, the true angular velocity of the ball, C_ω can be computed as $\frac{C(t_2)C(t_1)^{-1}}{t_2-t_1}$ (note that difference in orientation matrices is computed through inverse functions). When multiplied by the time of the flight, the result is the total cumulative angle truly rotated by the ball. We compute the same from $E(t)$ and ultimately compute the difference in the **Cumulative Angle Error (CUE)**. To understand the impact of higher spin, we also report the orientation error (OE) for varying angular velocity. For trajectory, the metrics are simpler, namely **Location Error (LOE)** and **Speed Error (SPE)** reported against various parameters.

3.6.1 Performance of Spin Tracking

(1) Cumulative Angle Error (CUE): Fig.3.18 reports the CUE for each of the 50 spin throws – the results are sorted in ascending order of cumulative angles. A Y axis value of 4000 implies that the ball has rotated $\frac{4000}{360}$, which is 11.1 cycles in air at the end of the flight. Evidently, iBall

Figure 3.17: (a) IR based ViCon cameras at the ceiling. (b) A cricket ball instrumented with IR markers.

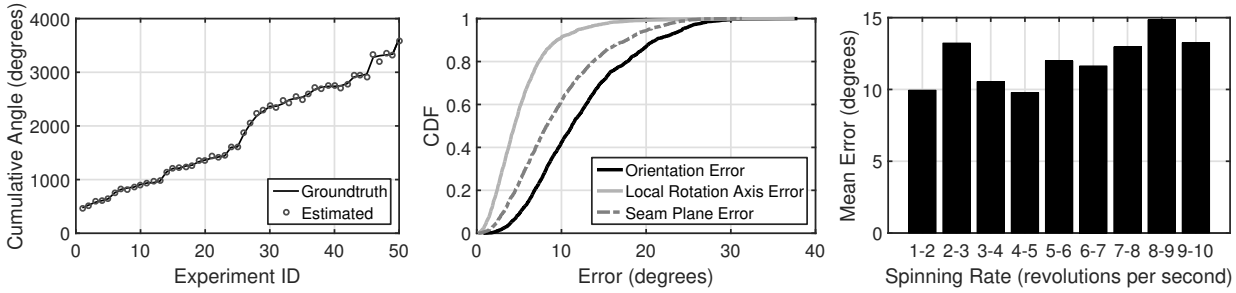

Figure 3.18: (a) Cumulative angle error (CUE) across different experiments. (b) CDF of orientation error (ORE). (c) Average orientation tracking error (ORE) under different spinning rate.

performs close to the Vicon ground truth for almost every throw. More importantly, unlike gyroscopes (which suffer from drift), the magnetometer does not accumulate error over time (since it measures the absolute North vector at every sample). This is a promising result and a valuable primitive for various types of ball analytics.

Error in estimated angular velocity (not shown) follows the same trend as Fig.3.18(a), since it is simply the cumulative rotation divided by flight time. Across 50 experiments, we observe median angular velocity error of 1.0% and a maximum error of 3.9%.

(2) Overall Orientation Error (ORE): Fig.3.18(b) reports the CDF of ORE across all 50 throws –

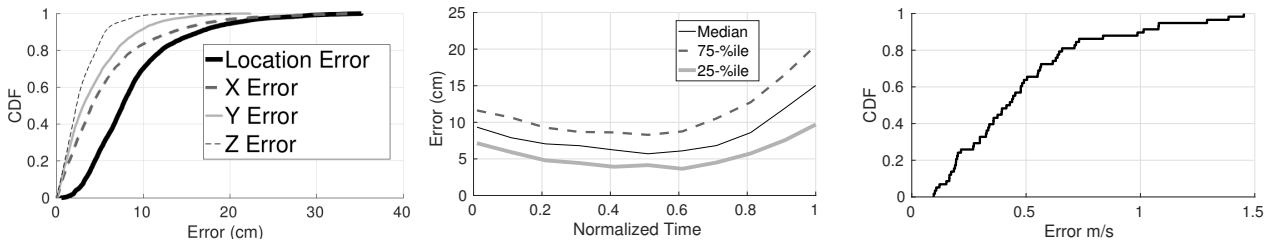

Figure 3.19: (a) CDF of location tracking errors. (b) iBall's location error degrades slightly toward the end of ball flight. (c) CDF of ball speed error.

the median is 11.2°. We also break down this error into local rotation axis error (AXE) and seam plane orientation error (PLE). We are especially interested in these two because accurately controlling rotation axis and seam plane is critical in maintaining the stability of the ball in the air. Results show a median AXE of < 5° and a median PLE of < 8°, while the 90^{th} percentile remains < 20°.

(3) Impact of High Spin: Fig.3.18(c) reports the impact of higher spin on ORE. The accuracy slightly degrades as the angular velocity increases. This is because, at higher angular velocity, our estimation of global rotation axis $\hat{R}^G$ is less accurate, degrading ORE. However, since the flight time is short, the accuracy degradation is marginal.

3.6.2 Performance of Trajectory Tracking

(1) Overall Location Error (LOE): Fig.3.19(a) quantifies the location error (LOE) across 50 different throws – the median error is $8cm$. We also report the errors on each of the directions: Y in the direction of the throw, Z being vertically upwards, and X is perpendicular to Y and Z. The median X, Y, and Z axes errors are $4.5cm$, $3.4cm$ and $2.39cm$ respectively. The X axis errors are maximum due to DoP effects, however, AoA lowers it to a reasonable value.

(3) Does LOE Accumulate at the End of the Flight? Fig.3.19(b) shows the median, 25^{th}, and 75^{th} percentile error for different positions of the ball during the flight. Importantly, since

we solve a global error minimization problem, the error does not accumulate. Still, the initial positions have higher accuracy compared to the end of the flight because AoA computed from the bowler–side anchor exhibits significantly less error. The degradation is still modest, with a median end-flight LOE of $15cm$.

(3) Speed Error (SPE) and Impact of Speed: iBall computes velocity estimates – Fig.3.19(c) shows a median speed error (SPE) of $0.4m/s$. Upon discussions with domain experts, we gather that this level of accuracy is valuable for coaching and analytics. Fig.3.20(a) decomposes the overall LOE results into different speed buckets. Evidently, the accuracy does not degrade at higher speed regimes (the maximum speed we could achieve in our experiments was $22m/s$). Of course, this is indoors and wind effects are minimal, if any.

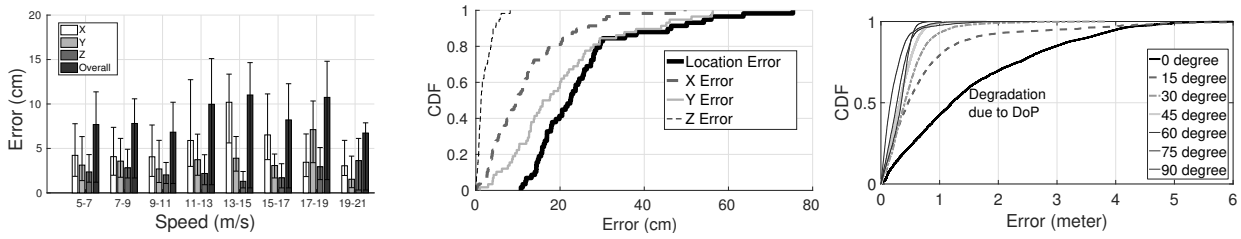

Figure 3.20: (a) LOE across ball speeds. (b) Prediction accuracy sufficient for LBW. (c) Player location error.

(4) Trajectory Extrapolation: The ball sometimes hits the leg of the batsman. An important question in Cricket is: *would the ball hit the wicket if the batsman was not in the way?* This is called *leg before wickets (LBW)*. For LBW decisions, its valuable to be able to predict (or extrapolate) the trajectory of the ball. The International Cricket Association has declared $10cm$ as the minimum tolerable LOE for LBW decisions. Fig.3.20(b) shows the trajectory prediction error with iBall. While the 3D LOE is $22cm$, the x-axis error is smaller ($9.9cm$), indicating the feasibility of using iBall for LBW decisions.

3.6.3 Performance of Player Tracking

Fig.3.20(c) shows LOE when the player is running at different parts of the playground – each line in the graph corresponds to the angle made by the lines joining the player and the two anchors. This also represents the LOE of the ball after it has been hit. We experimented in a real playground with a user running in a precise peripheral circle of $89m$ radius. At low angles (i.e., running almost collinear with the two anchors), the 90^{th} percentile error can be as large as $3.5m$. Our Kalman Filter based approach reduces this LOE to $2.6m$, while at higher angles, the LOE is already less than $50cm$.

3.7 Discussion and Future Work

(1) **Scaling to greater speed and spin:** Our maximum throw speeds were limited by our own abilities. Perhaps a bowling machine would serve as a better experimentation platform. For spin, limitations arose from ViCon – we observed increasing jitters and discontinuities in the ViCon data for spins above $12rps$. We are exploring alternative ground-truth estimation techniques at such spin regimes.

(2) **Indoor experiments:** We need experimentation under outdoor aerodynamic effects – the reported results may have been favorable in its absence. The lack of an outdoor ground truth system has been the bottleneck so far – we are exploring alternatives.

(3) **Multipath:** On the other hand, indoor environments may have affected the AoA estimates as well. In an outdoor setting, the multipath is expected to be less pronounced, potentially offering better reliability with AoA estimation and fusion.

(4) **Generalizing to other sports:** iBall's techniques extend to other sports with domain specific modifications. For example, stitches on baseball induce different aerodynamic properties than

cricket. We can incorporate the differences into iBall's mathematical models for tracking. Golf and tennis balls can be tracked too. Finally, iBall's techniques extend to hollow balls like soccer. Adidas micoach [104] has designed a soccer ball with multiple suspended sensors within the ball. This can potentially offer more information to iBall's optimization engine.

(5) **Smart ball weight distribution:** Our current prototype uses simplified instrumentation process for testing and alters the mass distribution of the cricket ball. However, the product version is expected to have a much smaller sensor foot-print, There is no need for a case or a battery (energy harvested from ball spin). Engineering experts from [92] have corroborated that a near ideal weight distribution (with stress tolerance) is feasible. In future, we will validate the weight distribution both quantitatively and qualitatively (feedback from players).

(6) **Enhancing Accuracy:** iBall's performance offers valuable primitives for various types of ball analytics. However, there are opportunities for pushing the accuracy levels further. We plan to jointly estimate the location and rotation instead of treating them as separate modules. The accelerometer can measure a combination of centripetal force (rotation) and linear acceleration. Similarly, the UWB ranging measurements contain a few bits of information about the orientation since the radios cannot be precisely placed at the center of mass of the ball. Such a coupling suggests that joint estimation of location and rotation can improve the accuracy of both. Hardware opportunities that leverage dual carrier UWB receivers can further decrease errors in AoA. While this comes at a slight increase in hardware complexity, we plan to explore the tradeoff.

(6) **Battery life and connectivity:** Our solution allows a battery life of $\approx$ 75 minutes between recharges, permitting short training sessions In future, perhaps wireless charging will mitigate this problem; perhaps fast rotation will automatically scavenge energy. Furthermore, the ball can be connected for cloud analytics through anchors. Sensor data from tha ball can be

streamed through its bluetooth radios and be analyzed, aggregated and compressed at anchors. Equipped with larger battery capacity and a longer communication range, the anchors can forward the data further for cloud based analytics.

3.8 Related Work

Embedded IMU: Authors in [105–108] embed IMUs in a Cricket ball and is perhaps closest to our work. However, these (brief) papers report basic features such as angular velocity, time of flight, etc. These features are directly available from the sensors and do not address the actual metrics of interest to the players/coaches. Authors in [109] also embed IMUs but focus mainly on the design and packaging of the ball for high impact. [110] explores spin-analytics in the context of a Bowling ball, however, due to low spin-rates and contact with the floor, accelerometers and gyroscopes are readily usable. This simplifies the problem in contrast to Baseball and Cricket.

Wearables, Cameras, and Sports Analytics: Several startups like Zepp, MiCoach, and Ball are extracting motion patterns from wearables. Smart sensor shoes have been proposed for analyzing soccer shots in [87], however, these are essentially classification problems. Hawk-Eye [111] is perhaps the most popular and expensive camera based tracker officially adopted in Cricket, Tennis, etc. Hot Spot [112] is a popular IR technology used to determine contact points between ball and players. Video analytics efforts in [113–115] are processing video feeds to learn/predict game strategies. While creative, the papers are addressing a different set of problems.

Localization and Motion Tracking: Rich literature in indoor localization [13–16, 116–120] has mostly focused on human motion. Under sparse WiFi infrastructure and high ball speeds, such techniques are inadequate. UWB based ToF ranging [121] report 10cm accuracy for static ob-

jects. We build on this technique but fuse with AoA, motion models, and DoP constraints, to cope with real-world challenges. On a similar note, inertial sensor based tracking have mostly been performed on humans, robots and drones [100, 122–126]. However, unlike iBall, none of these works address the space of freely falling objects. While work in [17] tracks ballistic missiles, the granularity of tracking is different both in time and space. iBall entails much finer granularities of tracking and appropriately formulates a global optimization problem for better accuracy unlike filtering techniques in [17].

3.9 Conclusion

This chapter develops techniques for tracking the 3D trajectory and spin parameters of a cricket ball. The core problem is rooted in motion tracking techniques, however, the sporting applications (and Cricket in this case) presents unique challenges and opportunities. Through fusion of wireless ranging, models of free-falling objects, and angle of arrival estimates, we formulate and solve error minimization problems. Results are promising and we expect our techniques to generalize to other sports. Our ongoing work is in pursuit of baseball and frisbee.

Chapter 4

The Case for Robotic Wireless Networks

4.1 Motivation and Vision

The last 30 years have witnessed significant advancements in wireless networking, ranging from hardware improvements to breakthroughs in theory, algorithms, and protocols. In the recent years, however, there is growing agreement in the research community that gains from the lower layers (MAC and PHY) are reaching saturation. Many are beginning to believe that the next "jump" in network performance will emerge from new ways of organizing networks [127–131]. In considering new network organizations, we explore the possibility of merging wireless networking with robotics. Specifically, we ask: *what if network infrastructure of the future – WiFi APs, enterprise WLANs, cell towers – are empowered with the ability to move physically?* In pursuit of this thought, we began surveying the current state of robotics, as well as the pros and cons of physically moving infrastructure (e.g., WiFi APs on wheels, or cell towers on drones). We make a few initial observations below.

(1) Infrastructure mobility may not be viewed as a one-size-fit-all solution, rather as a spectrum of opportunities illustrated in Figure 4.1. The opportunities range from centimeter scale *antenna mobility* to exploit multipath opportunities [132], to feet scale *tethered mobility* to evade wireless shadows and interferences, to full scale macro-mobility that minimize distance to clients. Network designers can choose to operate at different points on this spectrum, depending on user's requirements, budget, applications, and psychological comfort.

(2) Mobility is expected to bring a new degree of freedom (DoF) to network design, but more

importantly, this DoF compliments existing dimensions of wireless innovation. Techniques for power control, channel allocation, localization, topology control, can all benefit if APs are able to move, even in the scale of inches.

(3) The time scale of mobility can be regulated as necessary. Small scale mobility can be used to compensate for small changes in network conditions, while full scale mobility can be triggered occasionally when the system moves to a skewed state, or a strict QoS requirement is ordered. In cellular networks, for instance, quad-copters could occasionally fly out from cell towers and position themselves strategically to meet users' demands – like a network cloudlet [128, 129]. Infrastructure mobility could evolve as an on-demand service, a cost-effective and scalable alternative to over-provisioning.

Figure 4.1: Regimes of infrastructure mobility, ranging from centimeter scale micro-motions, to feet scale mini-motion under couches, to building scale macro-motion perhaps on tracks laid on ceilings. Further into the future, perhaps flying quadcopters can serve as cell tower extenders, parking at strategic locations to meet client needs.

Of course, some basic questions arise.

(1) Is moving infrastructure really practical? Concerns on feasibility are valid, but could perhaps be alleviated by building the vision in small systematic steps. Advances in personal

robotics, beginning from the popular Roomba [133] to the more recent quadcopters [134–137] are already mainstream. Hardware is rapidly becoming cheap and reliable – an Arduino based robot car chassis adequate for cradling WiFi APs is $16 today [138]. Based on where robotic technology stands today [139], it is certainly not the fundamental barrier to infrastructure mobility.

Questions on the architectural aspects are certainly more relevant, such as maintaining power/Internet connectivity to a mobile AP, tangling wires, awkward moving objects on the floor, etc. However, we do not envision an all-at-once technology deployment, rather we intend to activate functionalities incrementally. As a first step in home settings, a mobile WiFi AP might just remain tethered to power and Ethernet, and only move in small spatial scales (say, under the couch or study table). In enterprises, airports, and hotels, the APs may also be tethered, but they could move in a coordinated manner (like a joint topology control problem) orchestrated by the cloud. Moreover, the AP movements need not be continuous; the time scales could slowly become more frequent as the system matures and gains social acceptance. Of course, facilities management and other logistical/policy questions will arise, but we believe they can be mitigated if the core performance gains are compelling.

(2) How compelling are the gains? While the answer obviously depends on numerous factors, the high level message is that the *upper bound* can reach $3x$ and more, compared to the static case. For example, in home environments, median throughput from 2 feet of mobility is $2x$ for single clients, with the possibility of reaching $4x$ in 20% of the cases. With multiple homes, if APs coordinate to avoid mutual interference and optimize client SNR, median gain in *overall* network throughput can be $1.77x$ or more.

It is crucial to recognize that the performance gains are not obtained by moving the AP close to one client – with multiple clients associated to an AP, moving close to one client will adversely

affect others. The gains we observe actually arise from *finding appropriate AP locations from which the SNRs to all its clients are strong*. This is feasible due to rich spatial diversity in indoor environments, i.e., there exists certain nearby locations from which many clients experience strong channel conditions. In fact, the best AP locations could also experience lower interference from other APs and clients, enabling greater spatial reuse. On the other hand, blindly chosen AP locations can will fail to leverage these benefits, resulting in far inferior performance.

iMob demonstrates the ability to improve throughput to 5+ clients simultaneously. If too many more clients are active simultaneously, iMob can choose the *top-K* demanding clients and optimize their performance without affecting the others. If no solution is feasible, i.e., no AP location is able to satisfy the requirements, iMob could reduce the value of K. In the worst case, iMob will degenerate to a "static" AP and behave exactly as today's WiFi technology.

(3) Why move? Why not use MIMO, beamforming, or other software techniques? While these PHY layer techniques also leverage spatial diversity, mobility is still complimentary. Micro-shadowing scenarios are highly common in indoor environments [140, 141] – moving slightly can appreciably increase the rank of the channel matrix, resulting in higher MIMO gains. Our measurements confirm $3x3$ MIMO gains with today's 802.11 WiFi cards. Further, interference at the MAC layer is a function of energy, implying that AP1 would need to move out of AP2's *carrier sensing range* to enable spatial reuse. With beamforming/MIMO, AP1 will still sense AP2 and will defer communication. However, if AP1 could physically move out of AP2's range, or if AP1 and AP2 could jointly move to become "independent", system performance can improve further. Lastly, mobility and beamforming can be performed jointly to harness the best of both worlds.

The above is a high level vision (and qualitative arguments) aimed at motivating the overall research direction. We published a part of this vision in a workshop paper [142], along with

toy measurements on USRPs using 1 MHz frequencies. This chapter focuses on systematically characterizing the research landscape in real environments, and then builds a completely functional robotic AP system – iMob – using off the shelf 802.11n hardware. The key technical modules we develop are described next.

4.2 iMob: Robotic WiFi Access Points

As a first step of the broad vision, we focus on small scale mobility in homes, in a way that is minimally disruptive to the established notions of a WiFi network. The iMob system we develop will allow WiFi APs to move on wheels while being tethered to the same power and Ethernet cable, as is currently used in most homes. Ideally, the APs could be placed away from human movement, such as underneath a couch or a side-table, or at the corner of a room[1]. In this setting, the iMob system will be tasked to offer performance gains to client devices. The main technical components we develop are as follows:

• *We begin by measuring the upper bound on performance gain* achievable through feet-length mobility of WiFi APs. These gains are measured using a testbed of 8 laptops mounted on Roomba robots – the laptops run 3x3 MIMO using Intel 5300 802.11n cards. Using one of the devices as a mobile AP and others as scattered clients, we find the optimal AP location from which system performance is maximized. Besides serving as an *Oracle*, these measurements also offer insights into the nature of the gains, ultimately guiding the design of a real-time robotic networking system.

• *We cross-check the Intel card results with USRPs and Atheros cards* and verify that the gains scale across heterogeneous hardware (and not a function of our hardware idiosyncrasies).

[1] This is anyway the case in many homes, given that network devices and wires are typically hidden from eyesight.

• *We then develop a practical iMob system in which the AP observes channel conditions and moves in real-time to the best estimated location.* The motion planning algorithm uses insights from channel measurements, properties of the robot, and results from *optimal stopping theory*, to balance the tradeoff between exploration and exploitation (i.e., whether the AP should continue to explore more locations or should stop and perform remaining transmissions from its current location). This tradeoff naturally arises because the channel changes over space and time, and the AP does not possess the Oracle's view.

• *We also build a coordinated iMob system in which the cloud moves multiple interfering APs (e.g., in neighboring apartments or houses) to optimize performance.* This is essentially a topology control problem, with physical mobility as a degree of freedom. Both signals and the interferences can now be controlled to optimize desired performance metrics.

• *We evaluate single AP iMob in faculty homes, student apartments, and in our lab. Coordinated iMob is evaluated with* 4 *APs deployed across 2 floors in our engineering building.* Experiments are designed to evaluate a range of parameters and scenarios, including throughput and fairness, MIMO gains, impact of "leash length", impact of increasing number of clients, client mobility, etc. The overall gains are promising, and achievable without accurate prediction of wireless multipath and spatiotemporal channel variations. The inherent statistical nature of the environment offers viable opportunities.

4.3 Measurements

To characterize performance upper bounds with mobility, we will exhaustively move APs in small spatial granularities and pick the best location that optimizes a given metric – we call this the *Oracle*. We will then focus on understanding the nature of the gains, and utilize the insights to guide the design of a practical, real-time robotic WiFi system.

4.3.1 Experiment Platform and Methodology

Figure 4.2(a) shows a iMob AP assembled using a Roomba iRobot 2.1, a webcam, and a laptop equipped with Intel 5300 802.11n cards. The laptop is mounted on the iRobot and connected to it over the serial interface; it is also connected to a Microsoft live cam (attached in front of the iRobot) to guide its motion. The laptop acts as the controller for the whole system, sending motion commands to the robot (via the OSI interface), while also controlling the network interface for transmission/reception. 8 laptop clients were uniformly scattered at various locations and programmed to communicate back to the iMob AP.

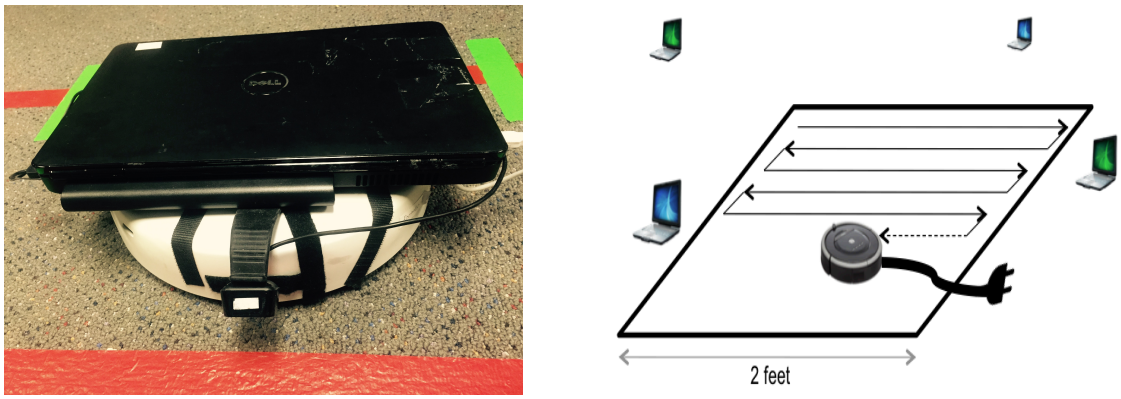

Figure 4.2: (a) A laptop and a webcam mounted on a Roomba to emulate a iMob AP. (b) Raster scan in a box while communicating to scattered client(s).

The robot's mobility is confined within a $2x2$ feet square region, demarcated by colored duct tapes pasted on the floor. If the robot drifts out of the square box, the camera detects the color of the duct tapes and triggers a change in heading direction. These square regions are selected from realistic areas in homes and apartments, i.e., near cable connection outlets. The AP performs "raster scans" within the square box (Figure 4.2(b)) at a speed of 10 cm/sec – during the scan, the AP continuously sends around 200 packets/second, equivalent to 60 packets per $3cms$. Transmissions are performed on regular OFDM with 3x3 MIMO at both 2.4GHz and 5GHz bands. Clients record the per-packet *channel state information* (CSI) for offline analysis [143, 144].

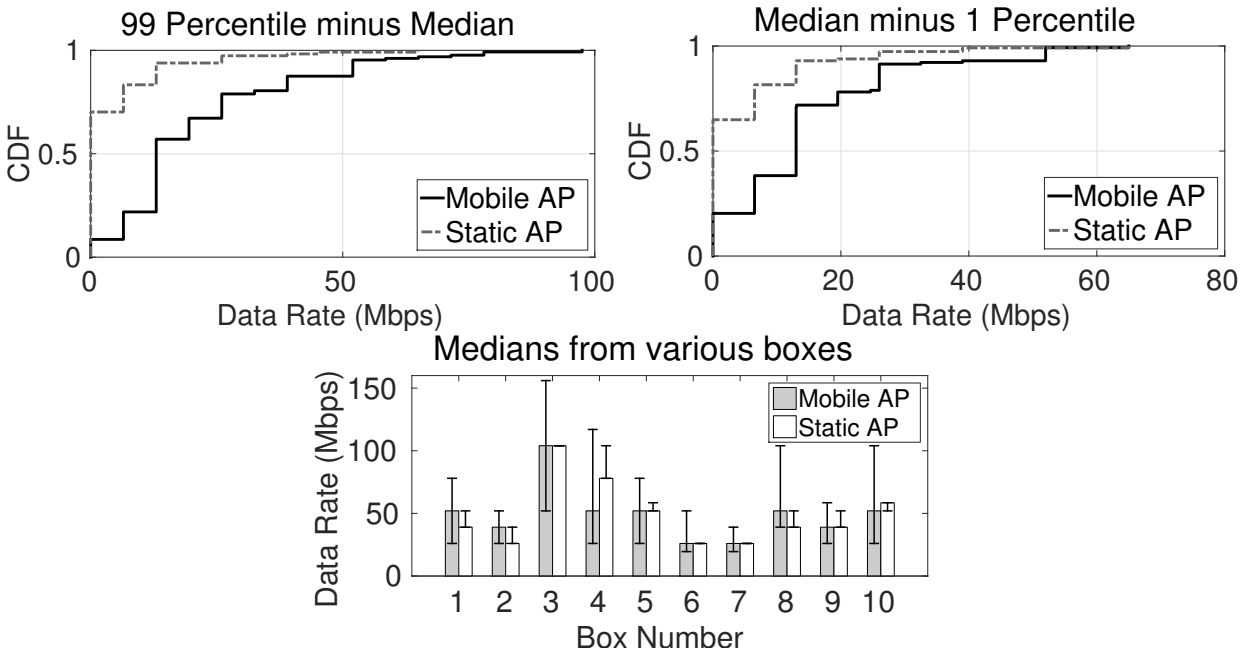

Figure 4.3: (a) Max minus $Median$ data rates for mobile and static AP, verifying the greater diversity caused due to mobility. (b) $Median$ minus Min data rates confirms that mobility also induces low data rates. (c) Comparison of the range of data rates for mobile and static APs in 10 randomly selected boxes (each bar representing $Median$ and error bars representing $[Max, Min]$).

The experiments were conducted in 4 different settings: (1) Student-office referred to as **Office**. (2) Various corridors opening into the atrium of the engineering building, called **Lab**. (3) Single bedroom graduate student apartment, called **Apartment**. (4) Large single family home with APs placed in different rooms, called **Home**. In all cases, people moved naturally during the experiments, and clients scattered at realistic locations. Total measurements exceed 100 hours, generating 5TB of data.

Metrics: We evaluate performance in terms of data rates, throughput, and fairness. While the *Oracle* selects the location with best data rate, our baseline scheme reflects today's static systems where the AP is placed at an arbitrary location near cable connection outlets. In light of this, the *median* performance among all locations inside the $2x2$ feet square is treated as the baseline. Thus, the upper bound gain, for throughput say, is defined as:

$$Gain = \frac{\max_{\forall i}\ throughput}{median_{\forall i}\ throughput}$$

where i denotes location i to which the AP can move to. Of course, when we design the real-time iMob system (later in Section 4.4), the median gain is not known to the AP since continuous raster scans are impractical. Still, the iMob AP should park itself at "good" locations from which the performance exceeds the median. We will discuss these later; for now, we focus on characterizing the system's upper bounds.

4.3.2 Characterizing Upper Bounds

The experiments are designed around 8 questions – the first 4 focussed on the amount of performance gain, and the next 4 on understanding the nature of the gains.

(1) How much Data Rate Gain at Single Client?

Consider a case where the iMob AP moves within a box while continuously transmitting packets, and 8 scattered clients record the *channel state information* (CSI) for every location of the AP. The CSI at each client can be accurately translated to the achievable data rate for communication between this client and the AP. For each tuple $< Box_i, Client_j >$, we compute the max, $median$, and min data rates (to avoid outliers, we always use the 99^{th} percentile as max and the 1 percentile as min). Figure 4.3(a) plots the CDF of max minus $median$ data rates due to the mobile AP, as well as the static AP, across all tuples. The key observation is that AP mobility induces large variations in data rates, far greater compared to the variations from temporal channel fluctuations. Figure 4.3(b) plots the CDF of $median$ minus min data rates for both mobile and static APs, and shows that the reduction in data rates are also equally stronger due to mobility. Figure 4.3(c) further compares the range of data rates experienced in the same box by a mobile and static AP – the error bars represent the max and min (Static's 1 percentile is sometimes the same as median due to low CSI variations). Clearly, mobility induces diversity.

While these results validate the known intuition that the wireless multipath signals interfere constructively or destructively in small spatial scales (causing diversity), it opens 2 specific opportunities for robotic WiFi applications.

(1) With centimeter scale mobility, an AP might appreciably improve data rate to a given client.

(2) With centimeter scale mobility, an AP can relocate to minimize interference from nearby APs/clients (potentially improving spatial reuse).

Assuming that the iMob AP is able to magically relocate to the best position, what is the gain possible compared to a static AP? Figures 4.4(a) plots the CDF of "rate gain" from 8 clients across 21 different boxes where the AP moved. We compute the rate gain as the ratio of $\frac{max}{median}$ data rate from each box. Evidently, an Oracle can easily *double* the data rate on average, and up to $4x$ in $\approx 20\%$ cases. Figures 4.4(b) now plots the CDF of "SNR reduction" to reflect how the mobile AP can move to avoid interference from nearby interferers. SNR reduction is computed as the *difference between median and minimum SNR* (note that interference is a function of energy and not the interferer's data rate, and hence plotted in terms of SNR). The achieved SNR reduction is around $4.5dB$ on average, contributing to a modest improvement in spatial reuse and throughput. In summary, the potential gains seem substantial given that the AP moved within a box of side 2 feet.

(2) Does Gain Scale to Multiple Clients?

In most realistic settings, the AP must serve multiple clients. So the natural question is: *is there any AP location from which the data rates can be simultaneously improved for all clients?* For this, we sum the data rates of all clients for each AP location within a given box – let S_i denote this sum for location i. Then we compute the average per-client data rate gain, β, defined as

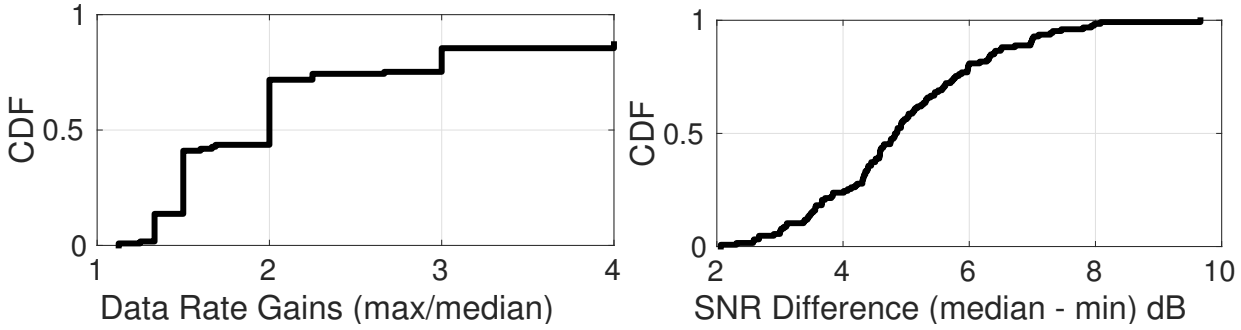

Figure 4.4: (a) CDF of $(max/median)$ data rates in a box indicates gain at a client. (b) CDF of $(median - min)$ SNR in a box indicates gain from avoiding interference.

$\frac{\max_{\forall i}\ (S_i)}{median_{\forall i}\ (S_i)}$. As before, the median represents the performance to be expected when the AP is placed statically at a random location.

Figure 4.5 plots the CDF of β for increasing number of clients. The gains are obviously expected to diminish since the AP must satisfy a stricter condition, nonetheless, the gains are still upwards of $1.35x$ on average even with 7 clients, and up to $1.45x$ for 3 clients. Homes mostly fall within this regime, where greater than 3 simultaneously backlogged connections are rare. In enterprises and hotspots (e.g., coffee shops), perhaps iMob can serve the 7 most data-hungry clients or the 7 weakest clients, improving the overall performance of the entire network. This result confirms the richness in indoor multipath diversity, offering support for robotic AP mobility even for the case of multiple clients.

(3) How much Gain in Throughput?

Figure 4.6(a) plots the CDF of throughput experienced by each client due to AP mobility. If an *Oracle* were to pick the best AP location, the throughput gain (compared to a random location) is shown in Figure 4.6(b). Aligned with expectations, the throughput gains are proportional to the data rate gains, although slightly less due to wastage from backoff and DIFS/SIFS slots.

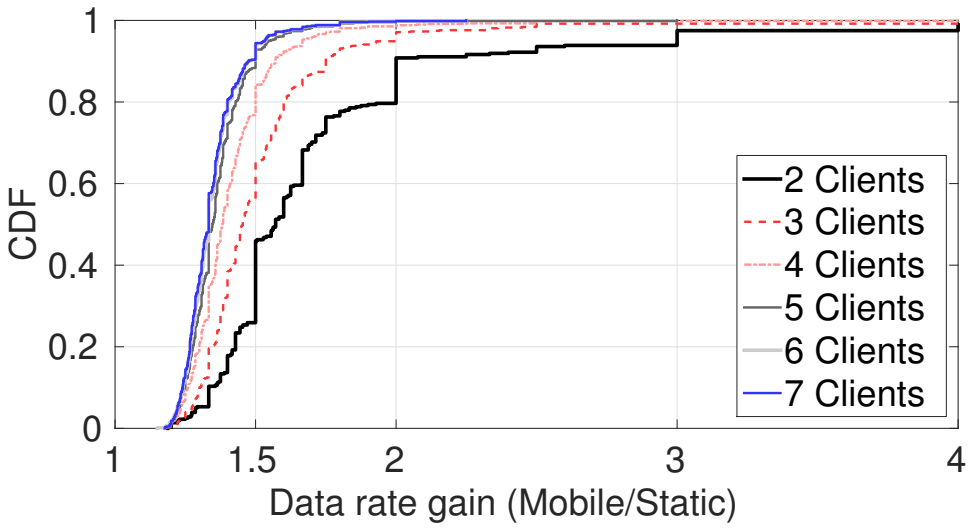

Figure 4.5: CDF of sum(data rate) gain over a static AP, where data rates are summed over multiple clients.

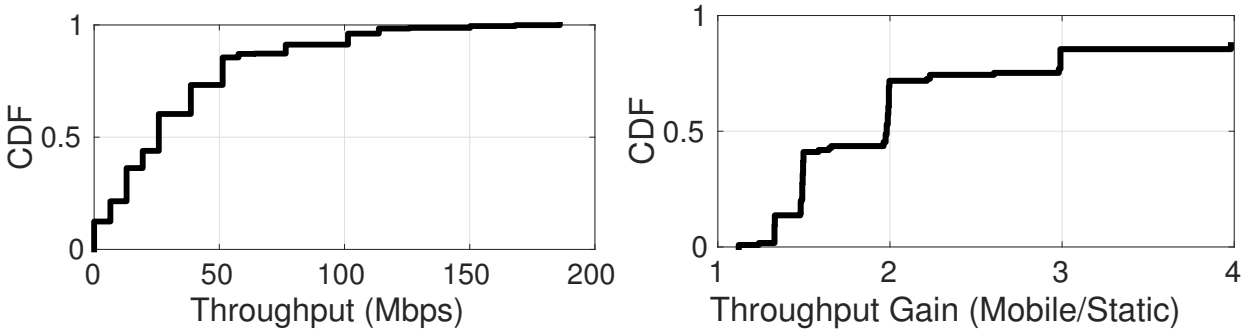

Figure 4.6: (a) CDF of throughput for each client during AP's mobility. (b) CDF of $(max/median)$, i.e., the Oracle's gain over a randomly placed static AP.

(4) Does the Gain Scale across Environments?

Figure 4.7 reports the *Oracle*'s median data rate gains from each of 4 environments, namely Office, Lab, Apartment, and Home. The reported gains are computed using the same metrics as above (i.e., max/median), and the experiments executed at 4 to 8 different places/rooms in each environment. The environment was entirely uncontrolled with natural human and object/furniture movements. Improvements are consistent, especially in the larger Office where the the clients are relatively further away from the AP (i.e., lower SNR). This is because modest

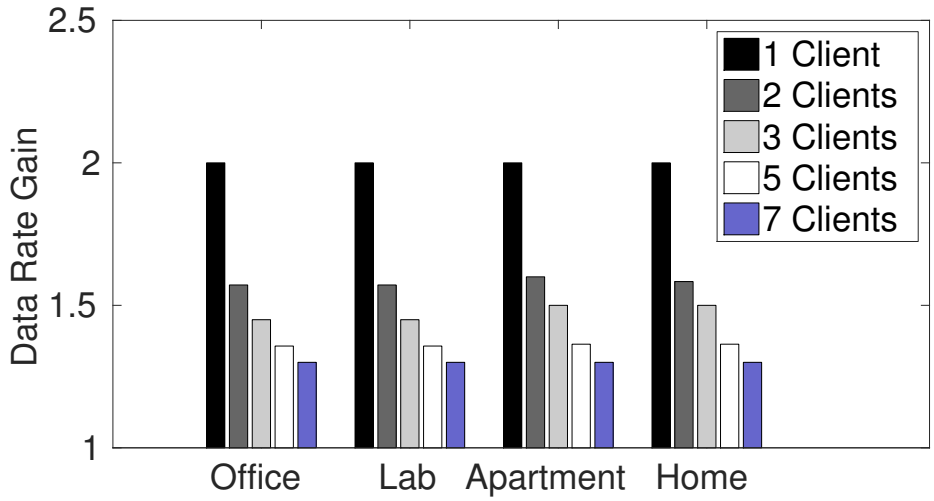

Figure 4.7: Oracle's median data rate gain in Office, Lab, Apartment, and Home in entirely uncontrolled settings.

improvement in SNR here can translate to greater rate improvement due to their logarithmic relationship.

To verify portability across hardware platforms, we performed similar measurements on USRPs and Intel cards. Figure 4.9 summarizes the results – this is loose in the sense that experiment conditions differed and some parameters were not identical (e.g., packet aggregation, MIMO, etc.) The key message is that the gains are consistent over static (single client), precluding any misgivings on our hardware.

4.3.3 Understanding the Nature of Gains

While the upper bounds on performance are valuable, the extent to which the bounds can be achieved is also important. The next 4 questions are focussed on achievability.

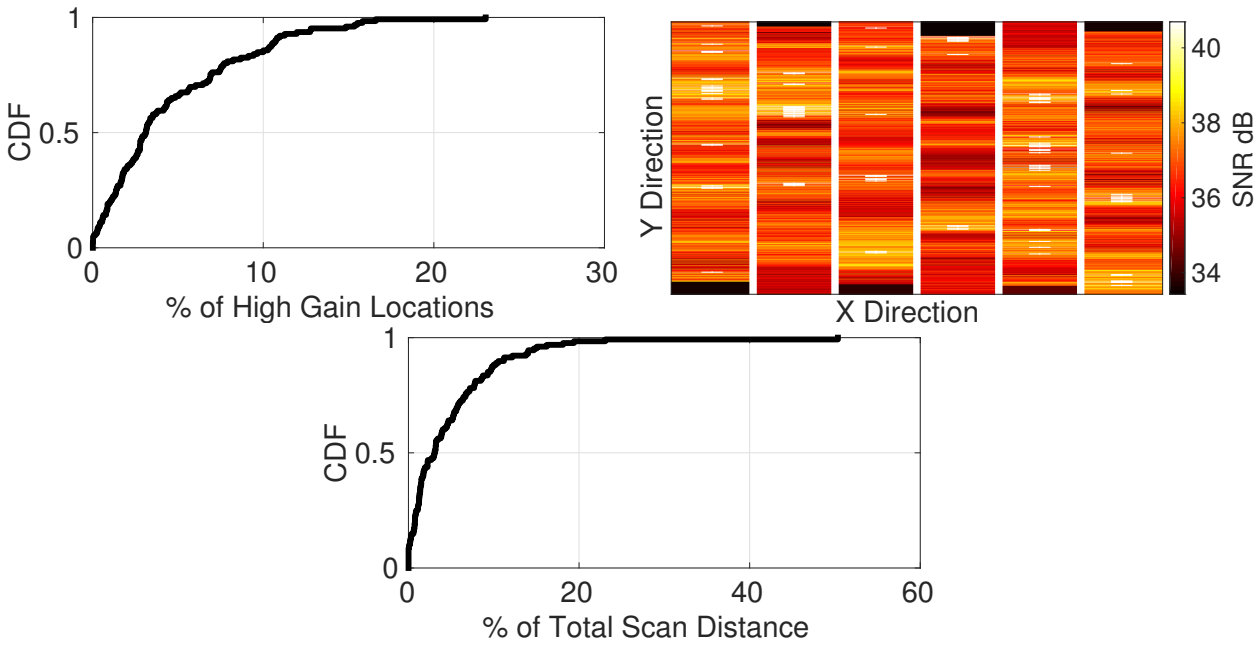

Figure 4.8: (a) CDF of number of high gain locations observed in one box (high gain defined as 0.95 of the max SNR in that box). (b) High gain locations within one box (marked with white dashes) shown as an example. (c) Distance traveled to reach high gain locations (distance defined as a percentage of raster scanning the entire box).

	USRP (Software Radio)	Atheros Cards (AR 5418)	Intel Cards (5300agn)
Interference Suppression	8 dB	6 dB	4.5 dB
Throughput	1.7x	2.15x	2x

Figure 4.9: Comparison across platforms.

(5) How Many High Gain Locations?

The existence of high gain locations is a necessary but not sufficient condition – if such locations are rare, the AP would have to spend a large time searching for it, affecting performance. Now, instead of targeting only the $maxDataRate$ locations, we define *high gain* locations as those that achieve greater than 0.95 times the maximum data rate in that box. Figure 4.8(a) plots the CDF of the fraction of these high gain locations, computed across 64 boxes from all experiments (we define "locations" as a $3x3$ cm^2 area as will be clear soon). Evidently, ≈ 40 high gain locations are available on average in a box, with some boxes offering far more. This is

a favorable indication.

(6) How Scattered are High Gain Locations?

It is important to also characterize the scattering of the high gain locations within the box – if all the high gain locations are clustered in a small region, searching one of them can still be time consuming. Figure 4.8(b) shows one example of the scattering in one box – the white marks denote high gain locations and visually illustrate that they are "well scattered". However, to quantify this, we compute the distance, δ, that an AP must travel to encounter at a high gain location. Figure 4.8(c) plots the CDF of δ with randomly chosen starting positions, and with mobility similar to a 2D raster scan within the box. Evidently, δ is quite small for a large fraction of the cases, suggesting that high gain locations can be encountered without searching for too long. This brings hope that the potential gains might actually be achievable.

Of course, the above graph also suggests that in some cases, the AP needs to move a large distance to encounter a high gain location. However, this does not mean that for these cases, the performance will be poor. To capture this, we attempt to answer the following question: *if the AP moves a pre-specified distance δ, what is the best performance that can be achieved?* Specifically, for increasing values of δ, we record the best data rate encountered, and compare this data rate against a static AP (i.e., median data rate in the box) and the *Oracle* (i.e., the max data rate in the box). Figure 4.10(a) and (b) plot the two comparisons, respectively – δ is defined as a fraction of a full raster scan in the box. Figure 4.10(a) suggests that even when the AP travels a small distance (δ = 5% of the raster scan), the data rate gain over static AP is still $1.5x$. Figure 4.10(b) suggests that this gain reaches close to the *Oracle*. Thus, the overall message is that strong locations are not elusive – even if the best location is unavailable, "good" ones can be found quite quickly.

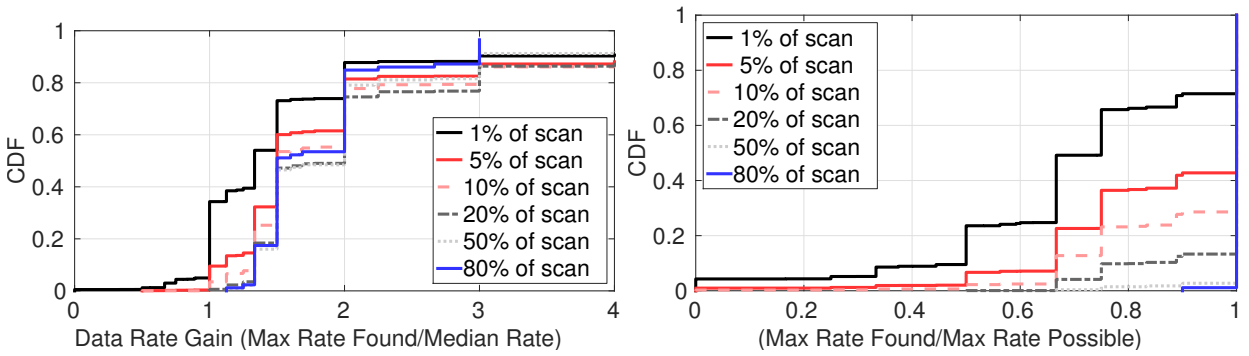

Figure 4.10: (a) The best location encountered after moving a small distance (local max) can still offer good gains over a static AP. (b) Local max is not too inferior compared to the *Oracle*.

(7) How Predictable are High Gain Locations?

In designing a practical system, it would be useful if the existence of a nearby high gain location can be predicted. Such predictions may be possible if the locations surrounding the high gain location form a gradient, like a "hill". On the other hand, if the surrounding locations exhibit significantly less correlation to the high gain locations, then predictions are difficult. To this end, we compute the CSI at a given location and measure how the correlation degrades as we move gradually away from it. If the correlation degrades gradually, it would indicate the "hill" we desire. Figure 4.11 shows the results of this experiment. Unfortunately, we observe that CSI correlations are strong until separations of $2.5cms$, but plummets drastically at separations of $3cms$ and more. This implies that the coherence region of a signal is around $3cms$, and locations outside that region is a poor indicator of its neighborhood. We term this $3x3$ cm^2 coherence region as a *pixel* – which now defines a "location" – and recognize that neighboring pixels will vary drastically in SNR or data rate. Thus, the data rate landscape is like a "jagged mountain range" in the granularity of $3cms$, making predictions difficult. These results and conclusions are consistent with multipath theory and independent measurements in literature [145–147].

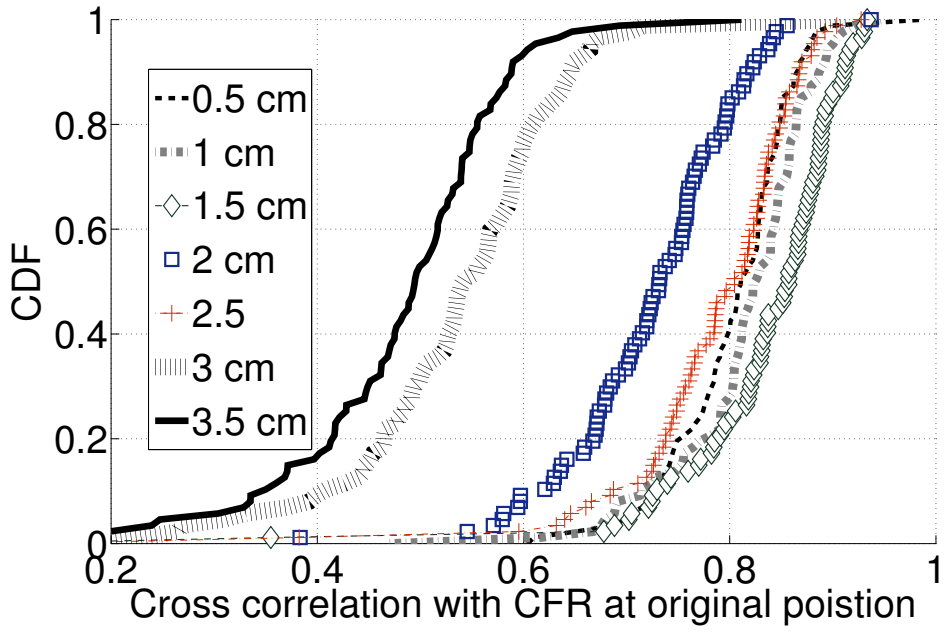

Figure 4.11: Data rates within a 5 cm shift of the mobile AP

(8) How Persistent are High Gain Locations?

If small changes in environmental factors cause the channel to change drastically, then iMob may not be worthwhile, since the AP will need to move very frequently. We classify environmental factors in 3 categories, namely human mobility, object mobility (e.g., doors, furniture), and device mobility (e.g., a smartphone moving in the user's hand). We then extensively investigate temporal stability by perturbing each of these factors – a human user typing on the keyboard, many people walking around, furniture moving, client laptops moving, etc. In the interest of space, we distill our key findings: (1) Client device mobility at the centimeter scale induces drastic change in the CSI, causing the channel to heavily fluctuate. iMob may not be beneficial to such devices (tablets, smartphones) when they are being held/carried in the hand. (2) For a static device (e.g., laptop, TV), human and object mobility impact the channel only when they block dominant signal components between the AP and the client. However, as shown in Figure 4.12(a) and (b), the channel revives once the human/objects have moved past. (3) Only when the human or object moves to a new position, and also blocks the dominant signals, the CSI

(and data rate) changes persist. However, such changes occur in the time scale of minutes [145] and can be detected by tracking changes in the CSI (detailed later). Thus, the take away message is that iMob could be effective even under dynamic environments, so long as the clients are static.

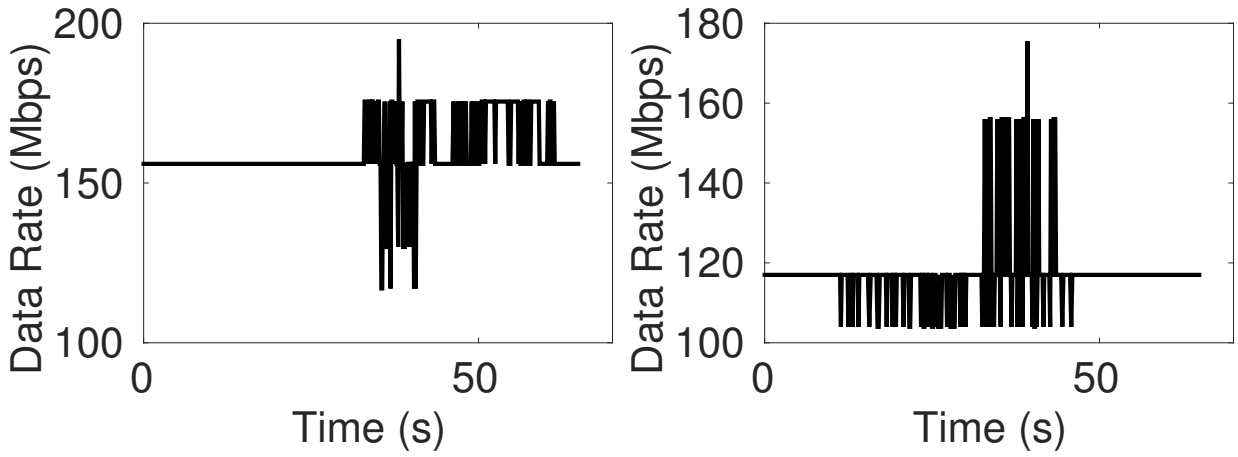

Figure 4.12: Data rate fluctuates when (a) humans, (b) objects go close to client, dwell for 10s, and walk past; the rate revives.

4.4 System Design

We take away 3 important messages from the measurements above: (1) The achievable performance improvement due to robotic AP mobility is substantial, available under realistic conditions (multiple clients and different indoor environments), and hence worth pursuing. (2) The high gain locations are challenging to model because they are randomly located, spatially small, and often juxtaposed next to poor SNR locations (making predictions difficult). (3) Although challenging, some opportunities offer hope – the high SNR locations are many, well scattered in a box, and stable for reasonable time scales even in real environments. This section is aimed at designing a practical AP motion planning algorithm that will suitably cope/leverage the above challenges and opportunities.

Some Design Guidelines

The core task of the algorithm is to search through different pixels (called *exploration*) and stop at a pixel that is expected to offer maximum performance gains (called *exploitation*). In the interest of space, we omit the various trials and deliberations that led to our final design; instead, we briefly discuss the key design guidelines that emerged from them. We will then assemble these guidelines into a practical iMob AP.

(1) Since AP mobility is at far slower time scales than packet transmissions, the *exploration* process must be speedy. Otherwise, an AP would spend unnecessary time at suboptimal pixels.

(2) Robotic motion is not accurate due to skidding of wheels, noisy compass values, mechanical turns – thus a robot cannot go back on the exact path on which it has traveled. This implies stopping decisions need to be made on-the-spot based on the SNR at that pixel. Performing a search and then retracing back to the max pixel on that path is not an option.

(3) The need to stop immediately at a high SNR pixel limits the maximum speed of the AP. Specifically, the inertial displacement after applying the brakes should be no more than a pixel width – to allow the AP to be within the same pixel once it decides to stop.

(4) Stochastic hill climbing or simulated annealing algorithms are not an option. Simulated annealing either incurs excessive time, or the starting point of the algorithm must jump to different random locations, which is impractical for the physically moving AP. Also, as mentioned earlier, these algorithms assume that backward motion is possible, which in our case is difficult.

(5) When clients move, or the environment changes too much, the CSI at the AP exhibits substantial change. This can be a trigger for the AP to re-explore the best pixel, since the current one may have become sub-optimal. This is particularly necessary when this client is

data hungry and optimizing its performance will boost the overall network performance.

Finally, and perhaps needless to say, the mobility heuristic must be lightweight to run on a simple robot in real time.

Optimal Stopping Theory

The crux of our heuristic is designed around a result from *optimal stopping theory (OST)* in applied statistics [18, 19]. The problem definition of OST is as follows. An employer intends to hire 1 individual out of n applicants (all of whom can be ranked based on quality). The applicants are interviewed one by one in a random order. However, unlike typical situations, in this case the interviewer must make a decision immediately after the interview; once rejected, an applicant cannot be recalled. Of course, during the interview, the interviewer can rank all candidates seen thus far, but is unaware of the quality of yet unseen candidates. OST asks: *which candidate should be selected to maximize the probability of recruiting the best candidate.* Selecting too early can leave many good candidates unseen; picking too late might mean that the best candidate is already rejected. The OST result dictates that the first $\frac{n}{e}$ candidates should be rejected, and among the subsequent candidates, the first on that ranks better than all $\frac{n}{e}$ candidates should be recruited.

OST bears a strong resemblance to our problem of selecting the best pixel, primarily because the pixels are scattered in an entirely random manner, with little spatial correlation ($3cms$) (Figure 4.11). Hence, there is hardly a notion of "gradient" to leverage. Moreover, channel modeling or ray tracing seemed impractical since the iMob AP does not have details of the environment (floorplan, furniture, etc.) that would influence the multipath signal components. A statistical approach seems inevitable. In fact, since high SNR pixels are not rare and quite well scattered (recall Figure 4.8(b) and (c)), a statistical approach may be able to find such a pixel within a short time. The time to search can be reduced by moving the AP fast during the *exploration* phase,

and slowing it down during *exploitation* (i.e., when its time to stop). With this background, we now describe the heuristic precisely.

Mobility Planning Heuristic

Figure 4.13 shows the flow-chart for iMob's mobility planning heuristic. The AP is placed at a random location by the user. Once it observes a stream of packets from a client, it begins an *exploration* phase. In this phase, it performs a raster scan at its maximum permissible speed, V_{max}, recording the channel state information (CSI) from each packet transmitted by client(s). Of course, the AP continues to communicate during exploration, moving through pixels of varying quality. The exploration continues until the AP has moved through $\frac{N}{e}$ pixels, where N is the total number of pixels in the box. At this point, the AP computes the best pixel among these $\frac{N}{e}$ pixels, where "best" is defined as an utility function of CSI:

$$U_{max} = \max_{p \in [1, \frac{n}{e}]} \left(\frac{\sum_i log(SNR_i)}{I_p} \right)$$

where p denotes a pixel covered by the AP, i denotes the index of its own clients. I_p denotes the number of interfering APs and clients sensed at p. The AP now enters the *exploitation* phase.

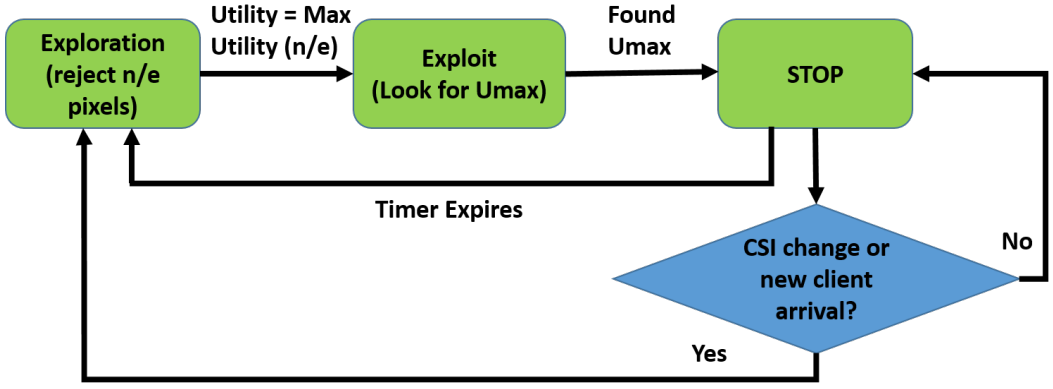

Figure 4.13: Core flow diagram of iMob's heuristic

During *exploitation*, the AP computes every pixel's utility, and stops whenever a pixel's utility is $\geq U_{max}$. However, to brake and stop in the same pixel, the velocity of the AP must be reduced during exploitation. Otherwise, inertia and skidding of wheels will propel the AP forward, and

returning back to this exact pixel will be time consuming. The reduced speed, V_{min}, is designed such that inertial displacement (after the application of brakes) is less than a pixel length ($3cms$) (discussed earlier). Once stopped, the AP continues communication with the client(s), expectedly at a near optimal data rate. The AP remains in this location until a new data hungry client joins, or until it observes a substantial change in the CSI of a client. Substantial CSI changes suggest mobility of the client or appreciable changes in the environment. Under both these conditions, the AP triggers the exploration phase again, and relocates to a new pixel. Additionally, the AP proactively relocates if it has been static for a very long time.

A common perception is that the exploration phase incurs a performance penalty because the AP is moving during this time and communicating from sub-optimal pixels. We observe that this sub-optimality is true with respect to the *Oracle* but not with respect to the static AP. Note that a mobile AP should statistically achieve the same performance as a static AP during exploration because the mobile AP will move through equal number of strong and weak pixels. Evaluation results confirm this (Figure 4.14(c)).

A natural question might be: *what if the channel quality at other locations improve over time – an iMob AP will not be able to proactively exploit this opportunity.* We observe that this is unlikely when CSI is used as the indicator function. If some other pixel has to improve substantially, then either the client must move to a new location, or the environment must change appreciably. Unlike SNR, both the effects will manifest in CSI variations.

Improvements to the Heuristic

We discuss a few optimizations to the core heuristic above.

(1) In some cases, the exploitation phase may not end quickly – the AP may not encounter a pixel offering U_{max} for a long distance. In such cases, the AP could be made to lower its expectations in proportion to the time spent in the exploitation phase. In other words, the AP

starts with the hope to achieve U_{max}, but progressively lowers the bar to some fraction of this value. The rational is stop *soon* at a pixel that offers reasonable utility, as opposed to paying the cost for finding the perfect pixel.

(2) Data hungry clients, such as those that perform video streaming, are likely to be the highest beneficiaries of iMob. However, most video streaming clients buffer data, leaving bursts of time in which packet downloads are much less. The AP could exploit these gaps to *explore* – if new pixels are discovered with greater utility, it could relocate. Recall that the pixel at which the AP stopped moving is not guaranteed to be optimal – its only a statistical estimate using OST. Exploring more can still be beneficial.

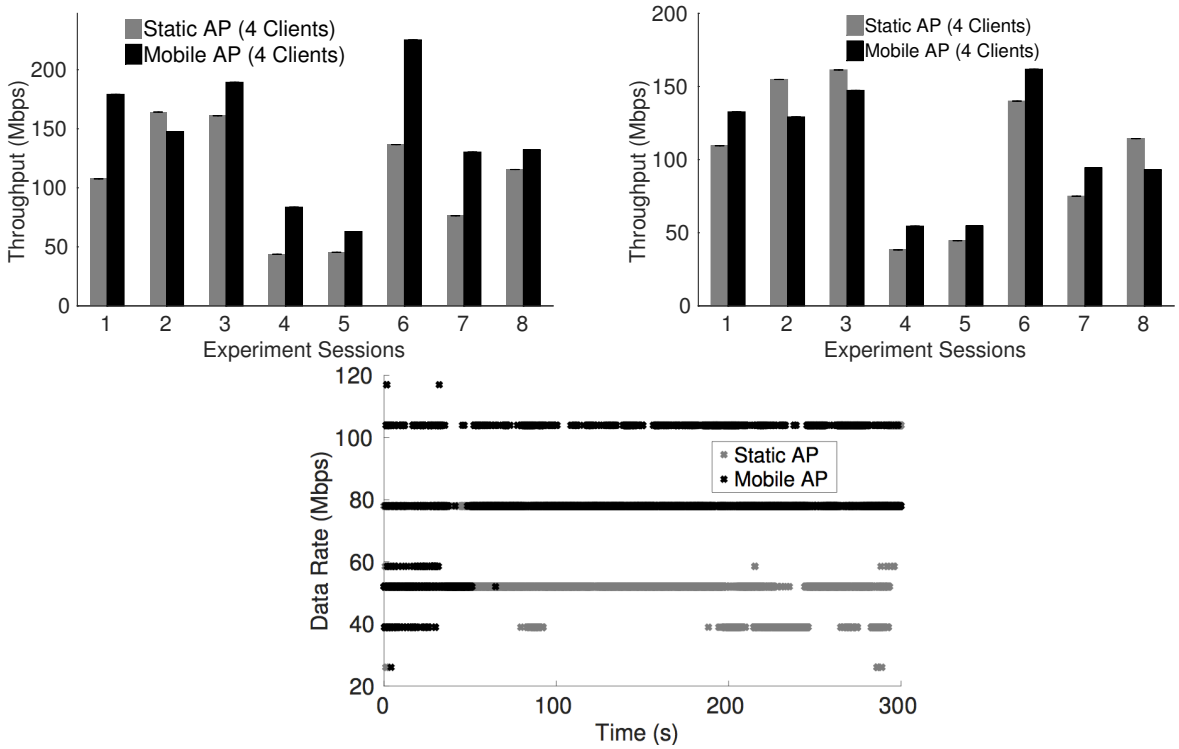

Figure 4.14: Throughput from real-time iMob with 4 clients: (a) Overall average throughput. (b) Average throughput when the AP is mobile, showing that AP mobility does not impose a performance penalty. (c) Data rate variation before and after stopping – the mobile AP's rates are comparable to the Static until it stops, and higher thereafter.

Multi-AP Coordinated Motion Planning

We extend the above heuristic to multiple APs (e.g., in residential neighborhood) by engaging the cloud as a mobility coordinator. For ease of explanation, let us assume K APs numbered from 1 to K. The cloud instructs the APs to enter the *exploration* phase together, and each AP computes the maximum utility among the first $\frac{N}{e}$ visited pixels. Of course, these pixels not only experience different SNRs but also different interferences caused by the other (simultaneously moving) APs. Each AP reports at the end of the exploration phase, and once all APs have completed exploration, the cloud again instructs them to begin *exploitation*. However, the exploitation phase is executed in series, meaning that AP1 performs exploitation first, followed by AP2, and so on. This partially ensures that AP_i has at least accounted for interferences from all $(i-1)$ APs. Each AP searches for a pixel that matches or exceeds its target utility, and stops upon finding one. Of course, this does not position APs in the optimal manner, but settles down in one reasonable configuration quickly. This also ensures convergence of AP movement.

4.5 Evaluation

We evaluate a completely functional single and multi-AP iMob system and focus on (1) the throughput and fairness comparison with today's static APs, (2) gap from the *Oracle*, and (3) impact of various parameters, such as client density, traffic sessions, mobility area, etc. We begin with a description of the methodology.

4.5.1 Implementation and Methodology

The evaluation platform is similar to the measurement platform, with the following key differences. The iMob exploration/exploitation heuristic has been implemented in the Linux kernel (Ubuntu 10.04) to completely operate in real time (e.g., pixel search, utility computation, Roomba speed control, braking). Performance is measured on the wireless link only –

the wired Internet connections at residences are the bottleneck, so connecting to the Internet would not reflect the actual wireless gains. We perform both single AP and multi-AP experiments. In the multi-AP case, a central server controls 4 APs – deployed across 2 floors of our university building – to extract holistic SINR and topological gains. Clients associate to our AP and upload/download packets over UDP/TCP while the AP moves to optimize performance. To compare against the Oracle, we performed experiments with continuous mobility and used the CSI data to precisely infer data rates [144] and throughput of each scheme. For realistic backlogged traffic, we record and use packet traces from YouTube, Google Hangout, and casual browsing sessions, captured from Wireshark. Across all experiments, the AP and clients were placed at realistic locations (to the extent possible). The environment was completely uncontrolled with people naturally moving, working, etc.

As a final point, Figure 4.15 plots the inertial displacement of our Roomba robot from the time of braking, for increasing AP speeds. Given pixels width of $3cms$, the maximum AP velocity prescribed by this graph should be less than 20 cms/s – we conservatively use 5cm/s since the braking may happen half-way into the pixel.

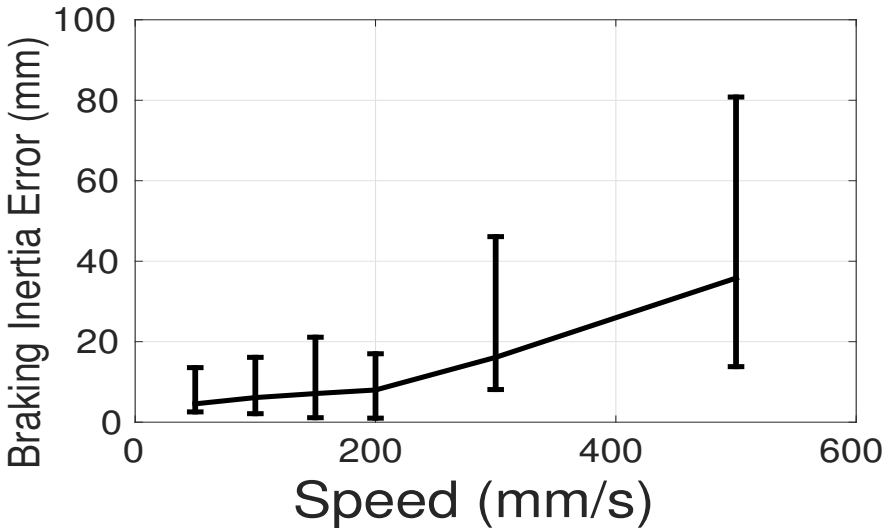

Figure 4.15: Roomba's inertial displacement after braking.

4.5.2 Real-time Single AP Experiments

Figure 4.14(a) plots the throughput comparison between iMob and a Static AP for various sessions, using 4 static and fully backlogged clients. Average throughput improvement is around 44%. One of the cases shows Static performing slightly better, perhaps because it was fortunately located at a strong SNR pixel. This is statistically a rare event, but possible.

Figure 4.14(b) compares the throughput achieved during the time the iMob AP was moving – this confirms that AP mobility does not impose a performance penalty. The throughput achieved by Static and Mobile are comparable since, statistically, the Mobile AP moves through both strong and weak quality pixels. However, once the AP stops at a strong SNR pixel, the performance exceeds Static thereafter, translating to net gain. Figure 4.14(c) zooms into the data rates observed during the *exploration* and the *exploitation* phase, showing how iMob's performance improves after stopping. Note that even while stationary, an AP (both Static and Mobile) still experience rate variations by around a notch due to temporal fluctuations (as seen earlier in Figure 4.3).

Coping with Environmental Dynamism

Observe that environmental dynamism will alter the optimal AP position, hence the iMob AP will need to trigger a new exploration phase. iMob uses a CSI based classification method that correlates the newly observed CSIs with recent CSIs, using techniques similar to [145]. If the correlation drops greater than a threshold, the AP triggers a relocation. For this, one of the clients was mounted on a Roomba and programmed to move periodically in our experiments – Figure 4.16(a) plots example timings of the client mobility and the Mobile AP's relocation trigger. The detection accuracy is robust and not affected by other humans moving in the environment. Figure 4.16(b) plots the detection accuracy across all experiment sessions, as a function of the distance the client moved from its prior position. In some additional cases, the AP also

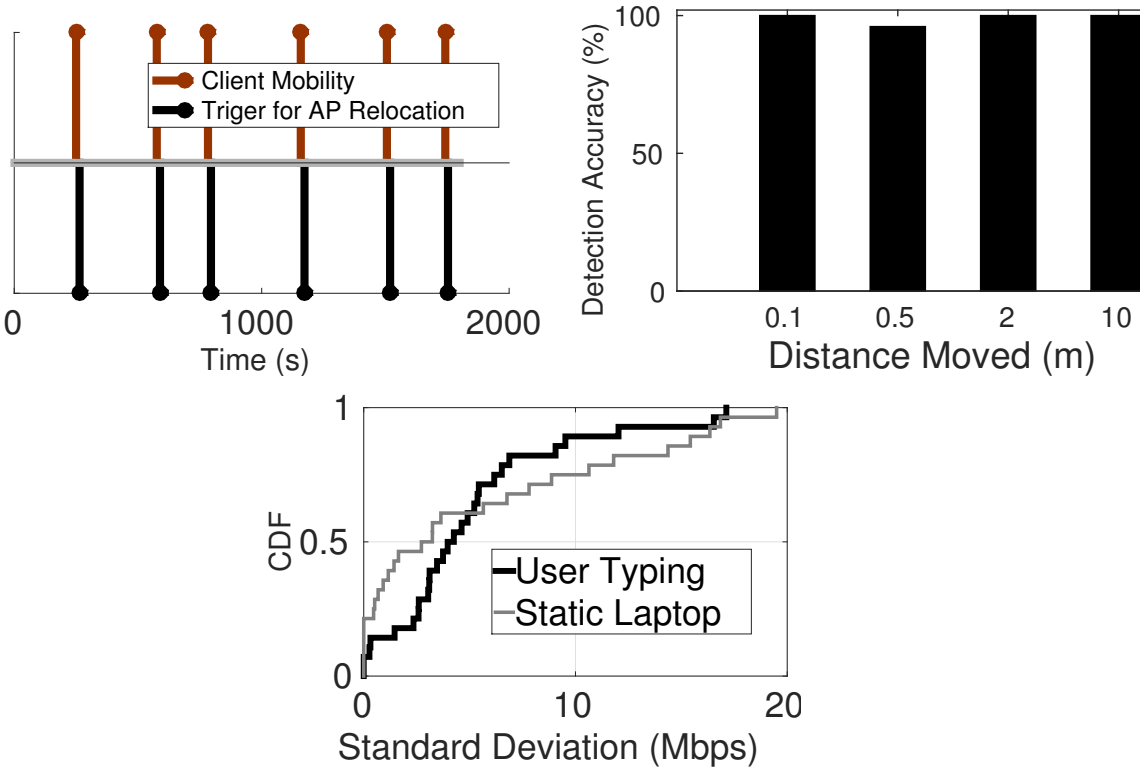

Figure 4.16: (a) AP detects when client moves and trigger relocation. (b) Detection accuracy for increasing client displacement.
(c) Variation of data rates when human typing on a laptop versus the absence of humans.

triggered mobility because of CSI changed (even though the client did not move), but we are unable to verify if it was a valid trigger. This is because we do not know the ground truth on whether the environment truly changed or not, hence false positives cannot be computed in such cases. To shed more light, Figure 4.16(c) shows the CDF of throughput variation between two cases: (1) a human is typing and working with the client laptop, and (2) the client laptop without the human user. The similarity in deviation suggests that the channel does not vary due to the human working, obviating the need for iMob APs to move in such realistic cases.

Fairness and Leash Length

Figure 4.17(a) shows that throughput improvements with iMob is not obtained at the cost of fairness. Using Jain's Fairness Index, we find comparable fairness performance as Static. Moreover, if desired, iMob can explicitly optimize for fairness, or even a combination of throughput

and fairness. Figure 4.17(b) plots the variation of throughput with decreasing coverage area of the mobile AP. The performance does not degrade too much, indicating that the diversity is truly rich. This bodes well for iMob – even where the AP has less than a feet to move around, the single AP throughput gains can still be 40%.

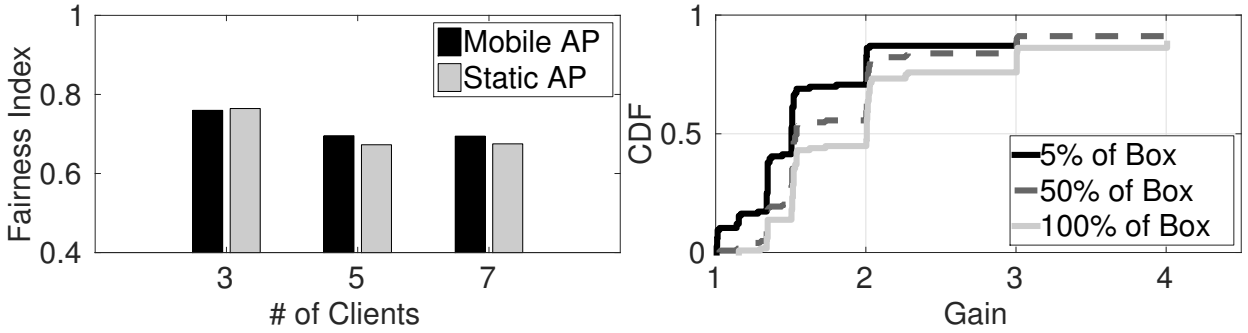

Figure 4.17: (a) Fairness does not suffer with iMob. (b) Throughput loss for decreasing mobility area.

Comparison with Oracle

Figure 4.18 compares iMob's performance against *Oracle* and Static AP, for single client scenarios. The experiment sessions are derived from wireshark traces of YouTube, Hangout, and a casual browsing session. For example, for YouTube, active time windows were concatenated, while intermediate gaps (typical for buffered playback) were not considered. Evident from the graphs, increasing session lengths improve throughput because the sub-optimality during the exploration phase gets amortized over longer session lengths, and the performance at the best pixel begins to play a more dominant role. Figure 4.18(a) shows that iMob remains reasonably close to the optimal, around 0.9. Against Static AP, iMob continues to achieve around 40% gain on average, but exceeds 80% in few cases of longer traffic sessions.

Figure 4.19 shows the variation of iMob's throughput against the *Oracle* and Static for increasing number of clients. iMob outperforms Static consistently and stays close to the upper bound. This suggests the efficacy of the optimal stopping heuristic to find a high quality pixel, even within 2 feet mobility.

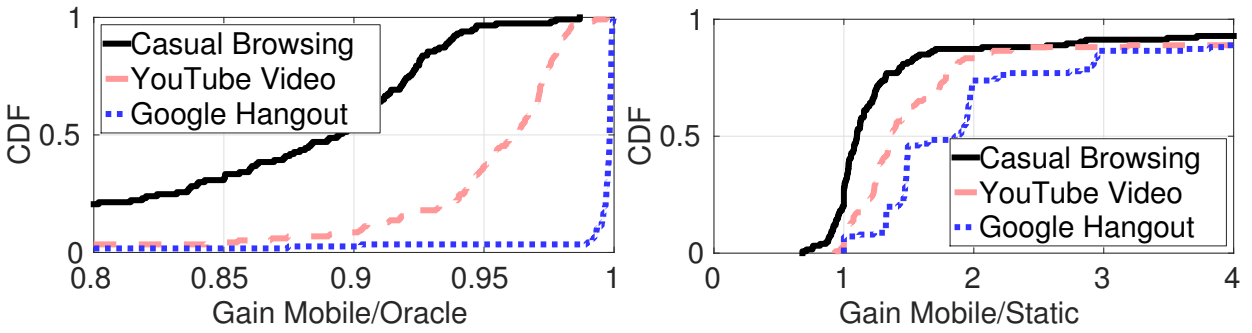

Figure 4.18: CDF of throughput gain for increasing traffic burst. (a) iMob over Oracle, (b) iMob over Static.

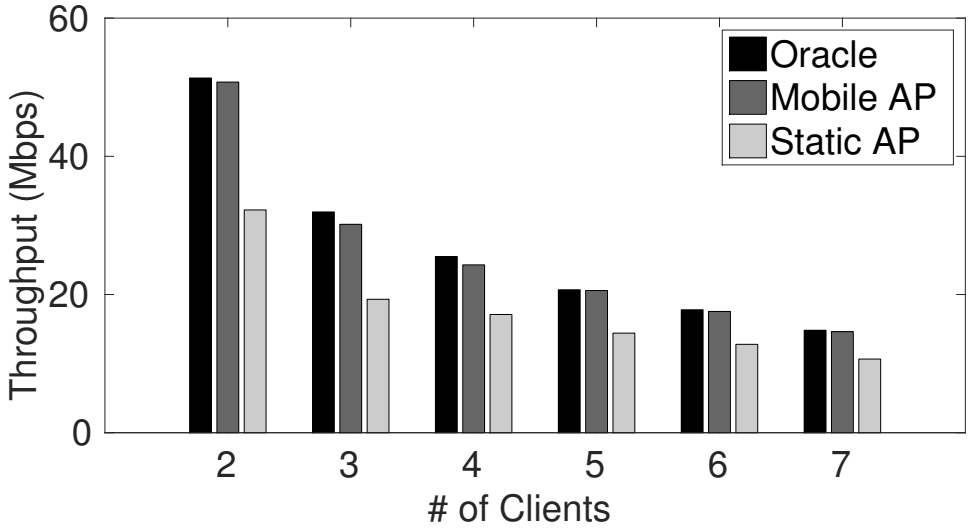

Figure 4.19: Median throughput for increasing clients.

4.5.3 Real-time Multiple AP Experiments

Figure 4.20(a) shows the topology setup in our engineering building. The testbed is spread over two floors (2^{nd} and 4^{th}) and consists of 4 APs with a total of 6 clients (each AP associated to 1-2 clients). All APs were placed in the 2.4GHz channel such that the neighboring APs are at the edge of each other's interference range. Transmit powers were assigned at 8dBm to all the nodes; clients remain static for all the sessions. The topology mimics an EWLAN network of access points where the APs in the *same channel* are placed far from each other. A central server connects to each AP over WiFi and coordinates their movements to configure an effective topology that offers strong SNR to the AP's clients but avoids interference (to the extent possible) from other APs.

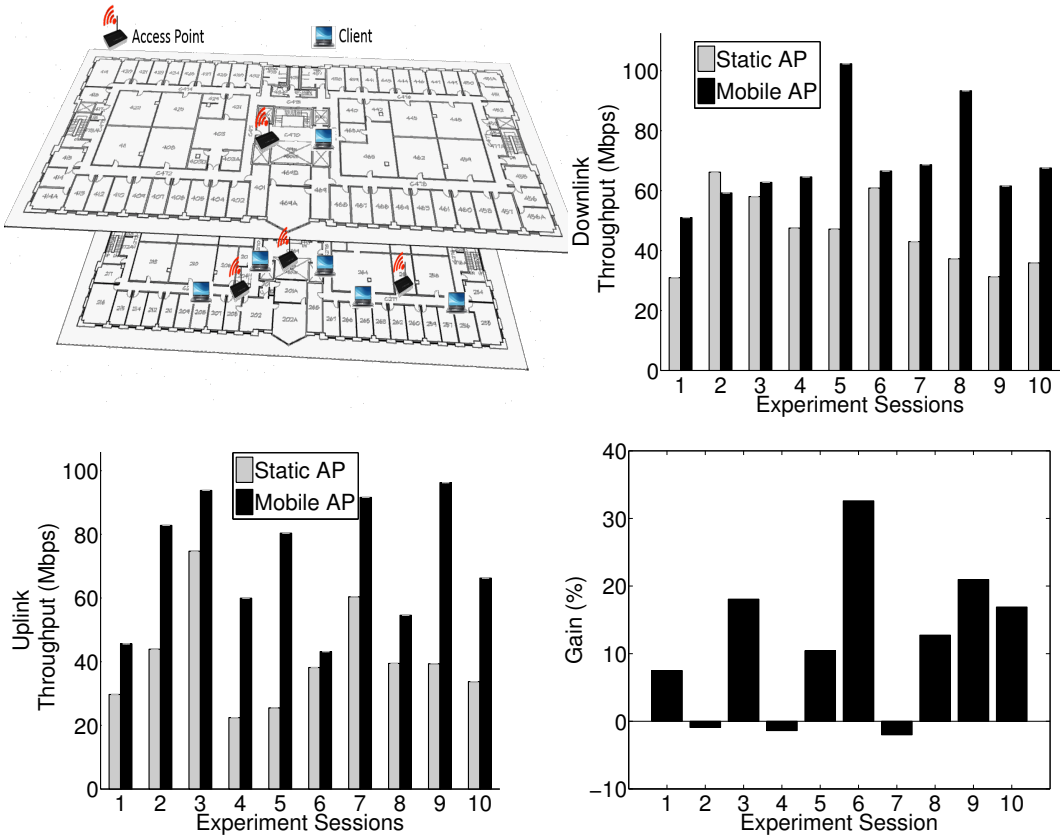

Figure 4.20: (a) iMob testbed deployed in 2nd and 4th floors. (b) Downlink throughput comparison. (c) Uplink throughput comparison. (d) Gain due to spatial reuse only, caused by sidestepping mutual interference from the other APs.

Figures 4.20(b) and (c) report the downlink and uplink UDP throughput comparison between the Mobile and Static AP. Gains are higher – 65% for downlink and 90% for uplink on average – implying that interference avoidance and better client SNR together contribute to net benefits. Fairness remains greater than the static case (not shown here). Figures 4.20(d) zooms into this break-up and shows the improvements due to spatial reuse. The "Gain %" on the Y axis shows how much extra opportunity was created by evading interferers in comparison with the static AP case. The average gain was about 12%, considerably less than client throughput gains. This is because of the binary nature of the carrier sensing threshold (APs need to find positions where the interferer is outside the sensing range). Nevertheless, the gains are still worthwhile because it combines multiplicatively with data rate gains resulting in net amplification in throughput.

4.6 Limitations and Opportunities

This is an early attempt to characterize and exploit the landscape of robotic wireless networks. Much remains to be done.

• **Moving client devices.** The key limitation with iMob is that constantly moving clients will not benefit from AP mobility, since the channel will change constantly. For such devices, however, the performance will still match the static AP. On the other hand, in favorable common scenarios where devices are static – video conferencing on laptops, streaming on smart TVs, even watching movies on a tablet on the table – gains are consistent.

• **Joint Mobility and Power Control:** Adding mobility to APs warrants revisiting classical problems in wireless networking. Power control and channel allocation can now be performed jointly with mobility, and adapted to changing traffic conditions.

• **Localization and Security:** Micro-moving APs may be able to mitigate the impacts of multipath, converging to a reasonably accurate pathloss index for their observed channel. Moreover, they could move macro distances to "look" at clients from different vantage points, ultimately improving the various techniques in triangulation and trilateration. Security benefits also emerge from mobility, thereby changing the channel properties that are used as the "secret key" between the transmitter and receiver.

4.7 Related Work

Closest to this proposal is probably *MoMiMo* [132], where the receiver adjusts its antenna in centimeter scales to perform interference alignment. While *MoMiMo* is a specific optimization for interference, this chapter attempts to create a broader theme of robotic wireless networks, and presents a case for the regime of feet scale full-device mobility. Perhaps a further step in this direction is "software defined mobility" where the cloud controls the mobility of network infrastructure. Finally, *MoMiMo* is complimentary to iMob – a WiFi AP can implement both.

Google Loon [148] provides Internet access to remote areas via ad hoc network–style balloons drifting above the stratosphere. DARPA envisioned the use of self-autonomous network of LANdroid robots [149] to provide connectivity in warfare areas. Our broad proposal certainly bears similarities, but focuses on injecting *controlled* mobility to today's established infrastructure.

Spatial diversity has been exploited in MIMO, beamforming [150, 151], and through other opportunistic ideas [152, 153]. Infrastructure mobility is by no means an alternative to these. Our results show that moving within a 2 feet box can yield higher data rates even with a 3x3 MIMO interface – we believe that feet-scale mobility can offer higher ranked channel matrices. From the robotics side, authors in [154, 155] have researched how robots cooperate to achieve a common wireless communication goal. In one instance, robots plan their motion paths to constructively beamform towards a specified receiver. Authors in [156] have envisioned robots forming a "chain route" to maintain connectivity to first responders (e.g., fire fighters) moving into a catastrophe stricken building. Delay tolerant networks have also considered node mobility [157], even in under water [158] and mobile sensor networks [159]. We believe this project is still different in the sense that it brings *feet-scale controlled mobility to existing network infrastructure that are conventionally viewed as static.*

4.8 Conclusion

This chapter envisions WiFi APs-on-wheels that move in controlled ways to optimize desired performance metrics. Early results are promising, although a deeper treatment is needed to fully characterize the interplay of many parameters underlying the success of such technology. Nonetheless, mobility is a valuable degree of freedom missing in today's network infrastructure, and extending research attention to it, we believe, is entirely worthwhile.

Chapter 5

Extending Cell Tower Coverage through Drones

5.1 Introduction

Outdoor cellular network traffic is steadily on the rise. Video is already the dominant application for cellular networks [160, 161]. In the near future, in-vehicle entertainment [162], uploads from numerous cameras, and various IoT applications (including smart cities, precision agriculture [163], and wearables [164]) will further add to the bandwidth pressure. Predictions indicate a 1000x increase in wireless data demand by 2020 [165, 166].

While technological advances in MIMO, beamforming, spectrum sensing, and others have coped with this pressure thus far, there is wide agreement that such opportunities are saturating. Users are beginning to experience spatial or temporal degradations in the quality of service. For instance, areas with tall buildings are suffering from poor SNR due to wireless shadows [167]; flash crowds at political rallies, sports events, and other social occasions are creating sudden traffic spikes [168]; natural disasters are destroying local network infrastructure, warranting a quick and temporary replacement service. Solving these problems with additional tower installations does not scale—the cost of over-provisioning is becoming excessive, exacerbated by the difficulty in finding installation sites in dense urban regions. This chapter takes an exploratory step and envisions drones as "elastic extenders" of cell towers. We call our system *DroneNet*.

DroneNet's model of operation bears similarity to cloud computing. Clouds leverage statistical aggregation opportunities, i.e., given that only a fraction of clients are requesting resources at a

given time, the total resources in the cloud need not scale with the number of clients. Yet, any given client can still avail powerful resources from the cloud. Drones bring similar flexibility to the wireless world. Not every user experiences dead zones at a given time; neither are all users located in a flash crowd. Moreover, traffic demand exhibits a power law behavior; few users form the majority of demand at a time. Hence, a limited set of drones may be adequate to address all the dynamic needs, leading to a win-win situation for both the clients and network service providers.

Figure 5.1 illustrates a toy example. It shows multiple possible locations at which the drone can hover. From any of these locations, the drone connects to the clients with a WiFi link, while the backhaul operates over a 4G/LTE link back to the cell tower. As mentioned earlier, the core research question pertains to determining the best hovering position. Observe that moving closer to the ground improves client proximity, however, the multipath and shadowing effects get severely exacerbated. Moreover, the line of sight (LOS) to the cell tower also gets disrupted. Moving vertically higher offers better LOS to clients and the cell tower, but at the expense of longer distance to these clients, reducing data rates. Lateral movements also pose tradeoffs – for instance, the left and right-most positions in the Figure 5.1 both offer LOS paths to one of the clients but blocks the other. A combination of lateral and vertical movements can bring the drone to a position that maximizes a given function of SNR. Our goal is to efficiently determine this location.

A brute force solution would be to fly the drone and conduct SNR measurements to empirically search for the optimal location. However, a large 3D search space – say 2 or 3 city blocks in Chicago – makes this approach prohibitively time consuming. Movements in clients and changes in traffic patterns will occur at faster time scales, rendering this brute force search useless. Hence we require a solution that is lightweight and quick. Simple strategies like hovering at the centroid of a group of clients are unsuitable due to the non-monotonous relation

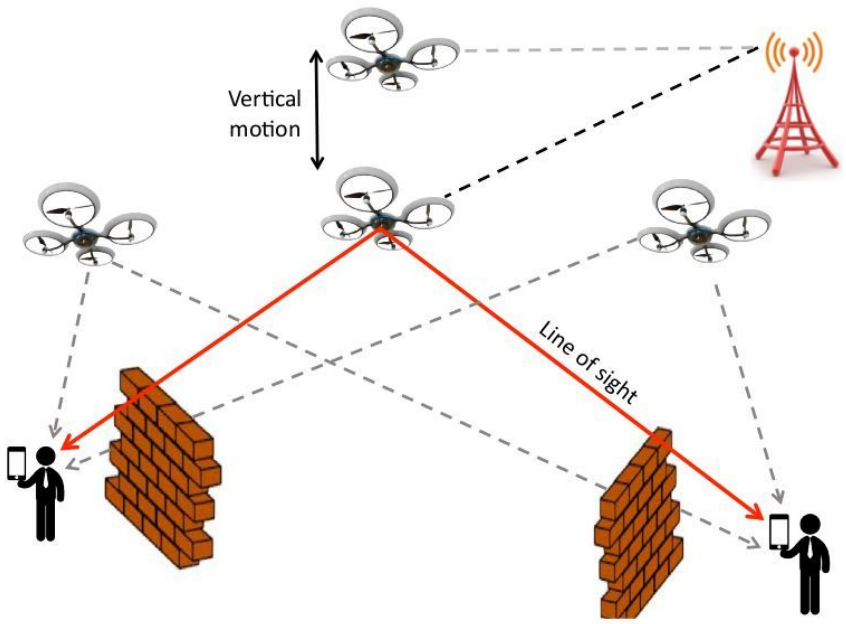

Figure 5.1: Drone locations present tradeoffs: Closer to the ground aggravates multipath and blocks line of sight (LOS) to the cell tower, while vertically higher placement increases distance to clients. Lateral movements can achieve LOS to some clients but get blocked from others.

between distance and SNR. Results from historical searches are also not useful since each new situation is somewhat unique in its placement of clients and the type of traffic demands.

We explore the possibility of using RF ray tracing as a hint to narrow down the scope of search. Our key idea is to model the dominant structures located in an area—such as the buildings and trees—to roughly model the terrain of the region, and then simulate how signals would bounce and scatter from the drone to the various clients. Of course, such simulations yield coarse-grained results since the simulated SNR is sensitive to centimeter-scale errors. Nonetheless, we find that these simulated results can still be valuable in broadly guiding the drone towards the right direction, i.e., towards areas where the SNR is relatively better. Once the drone arrives in this area, it physically conducts measurements to fine-tune its hovering location. Given a considerably smaller search space, the operation incurs far less time. The drone now hovers at this location offering connectivity to clients. If client positions or traffic changes substantially,

the drone recomputes the ray-tracing results and finds a new hovering location.

Early prototype with 7 clients spread across an area of $160 \times 280m$ in the UIUC campus shows promise. We conduct flights with an octocopter, carrying an Almond WiFi AP and continuously transmitting packets to clients. The octocopter moves in a raster-scan over an area of $50 \times 68m$, and at three different heights at 15, 30, and $45m$, above ground level. We conduct ray tracing simulations using Remcom Wireless Insite [169]. We observe consistent correlation between the ray-tracing model and measurements – *DroneNet* exploits this to extend 44% throughput gain with only 10% of full measurement overhead.

We briefly summarize the contributions as follows:

(1) *Exploiting inaccurate ray tracing as an opportunity to reduce the search space for drone placement.* Fine tuning the SNR search through small scale physical movements of the drone.

(2) *Measurement based evaluation on a real drone, flying on the UIUC campus while communicating to 7 clients on the ground.* Significant engineering effort towards building this infrastructure, including payload optimization, battery management, data rate selection, power control, ground truthing, etc.

The rest of the chapter expands on these ideas and experimentation effort. However, we first address some of the natural questions and discuss other alternatives to solve the problem.

5.2 Natural Questions

Does today's battery technology support longer flights? Flying a drone requires significant battery power. However, constant hovering may be unnecessary. Drones may hover only under extreme situations, while under less dire needs, they may fly and park at charging stations on top of buildings, lamp-posts, or fences. Gas/solar powered drones offering extended lifetime

are also becoming available [170]. Alternative designs based on balloons consume relatively lesser energy [171]. We are aware of current battery limitations, however, we see opportunities emerging in the future that will make extended drone flight viable.

How practical is obtaining terrain model? Public data from Google warehouse is available for many buildings [172]. However, it is possible to create 3D models of an area by taking pictures from drones and using photogrammetry techniques [173]. We do not rely on material of the terrain.

How compelling are the gains? The gains are promising. Figure 5.2 shows that 16-18dB of SNR gain is possible from drone mobility at various heights (30m, 45m, and 60m). This translates into a throughput gain of 44%.

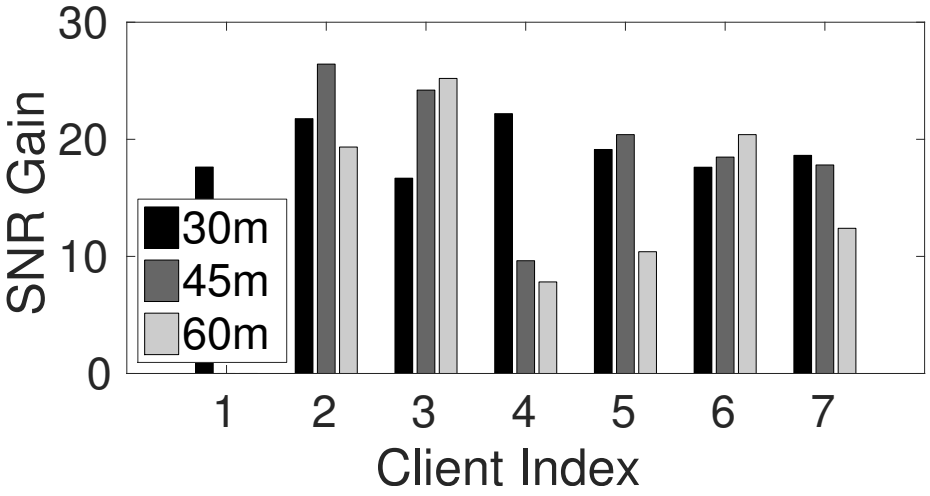

Figure 5.2: SNR gains for various heights across all clients

Are the gains emerging from moving closer to clients? Moving closer does not necessarily improve SNR due to multipath. To elaborate more, suppose that drone mobility is constrained in a region R. Let P_{close} denote a location within R that is closest to a client. Now, let P_{10} denote a subset of locations within R such that the SNR to the client from these locations is ranked in the

top 10 percentile. Figure 5.3 shows the separation between P_{10} and various locations in P_{close}. Evidently many high SNR positions exist when the drone is 20-30 meters far from P_{close}. This suggests that the drone need not go closer to achieve a near-optimal SNR.

Figure 5.4 shows the best SNR in R for different heights of the drone at various client positions. SNR can both decrease or increase with height depending upon terrain structure, client location. SNR variation is non monotonous and going closer to the client by decreasing the height is not always beneficial.

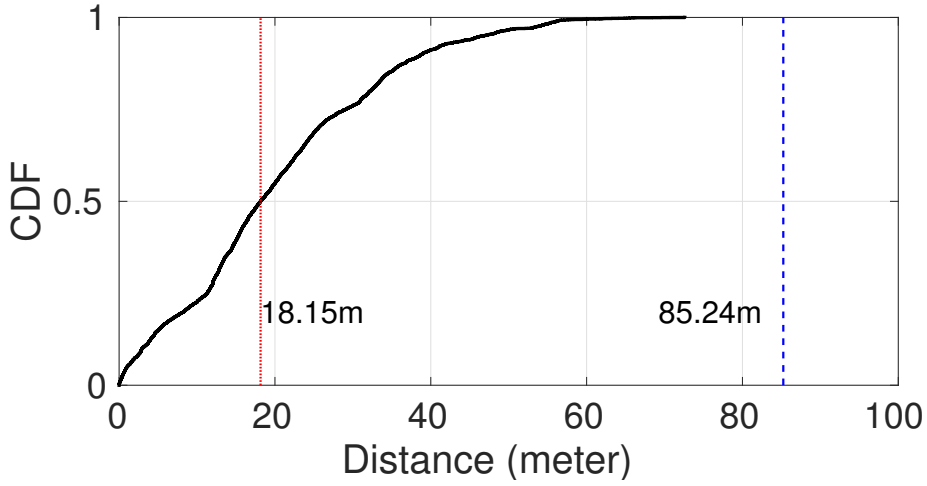

Figure 5.3: High SNR positions are scattered everywhere

Why not use Femtocells or WiFi Hotspots? While beneficial, static infrastructure entails high density to simultaneously cover all shadows. Moreover, the height of deployment is limited to height of buildings. A drone on the other hand can fly higher and offer much higher coverage. The drone's motion allows on-demand mitigation of dead zones and hotspots.

5.3 System Design

Figure. 5.5 summarizes the flow of operations in *DroneNet*. Given a set of client locations and the terrain model, *DroneNet* first runs a low fidelity, light weight ray tracing simulation

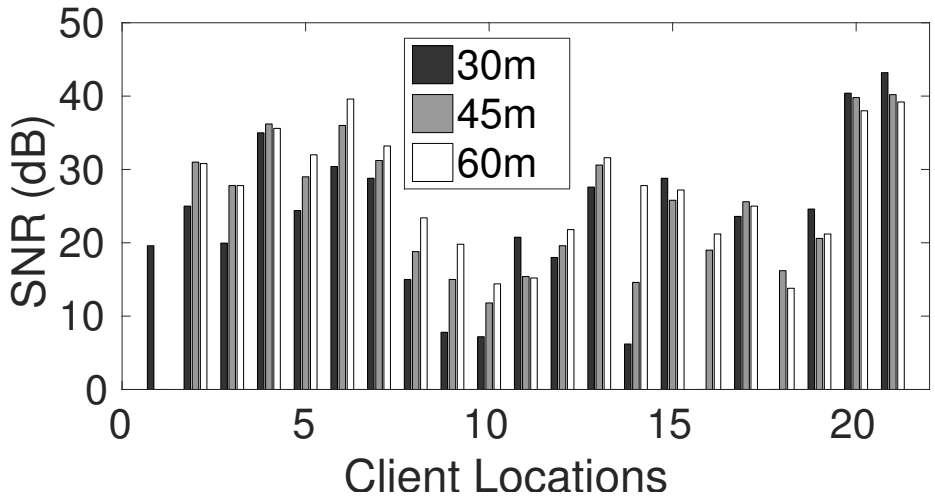

Figure 5.4: Best observed SNR for different drone heights

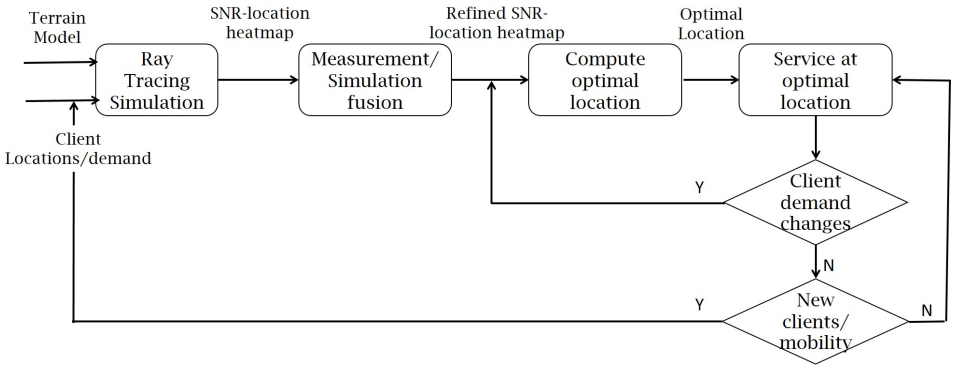

Figure 5.5: Flow of operations in *DroneNet*

to compute SNR at each client as a function of drone location. A 3D heatmap showing this SNR for each client is obtained. The SNRs are translated into throughput using Shannon's equation[1]. The sum throughput over all clients is computed, resulting in a 3D heatmap of aggregate-throughput as a function of drone location. *DroneNet* then conducts a quick scan of physical measurements around the regions of high throughout in the heatmap. During the scan, the drone finds the position offering maximum aggregate-throughput across all clients and hovers there. Whenever the traffic demand of clients changes, or clients move, *DroneNet*

[1]We consider throughputs of only client-drone links (without including the drone LTE backhaul) since they create bottleneck. The LTE backhaul is expected to have clear line of sight to the basestation, therefore of high quality. However, client-drone links are more challenging due to terrain shadowing.

should adapt accordingly. However, we leave addressing client mobility to future work. We now expand the two key modules enabling *DroneNet* design: (1) Ray tracing simulation, and (2) Fusion of simulation and measurements

5.3.1 Ray tracing simulations

Using ray tracing simulations, *DroneNet* predicts SNR heatmap without undergoing the overhead of empirical measurements. A basic "shoot and bounce" ray tracing [174] simulation is conducted using Remcom Wireless Insite software [169]. We describe the steps involved using Figure 5.6 as an illustration. It shows a few buildings, one client, and an area where the drone can potentially fly. The goal is to predict SNR from ray tracing.

1. Many rays pointing in all directions are generated from the client position. Figure 5.6 depicts rays emitting from *Client-1*
2. Each ray advances, and hits objects in the environment such as buildings and trees.
3. This causes deflections—reflections and diffractions. While each reflection creates one reflected ray, diffraction creates multiple rays. The deflection of rays generated from *Client 1* is observable in Figure 5.6.
4. Appropriate amplitude is set for deflected rays based on propagation delay and the absorption, reflection and refraction coefficients of incident material. We assume a single material for all structures, resulting in some inaccuracies.
5. Rays with weak signal strength are eliminated. This includes: (1) Rays undergoing more than 4 successive reflections. (2) Rays undergoing more than 3 successive diffractions. (3) Rays traveling longer than a certain threshold.
6. Finally, all rays arriving at the drone location are noted. The SNR is computed as a function of amplitude and delay of the arriving rays. Figure 5.6 shows the set of rays converging at the drone location *L1* using which the SNR at *L1* can be computed.

Similarly, SNR of all other possible client-drone links are computed to obtain their own SNR heatmaps.

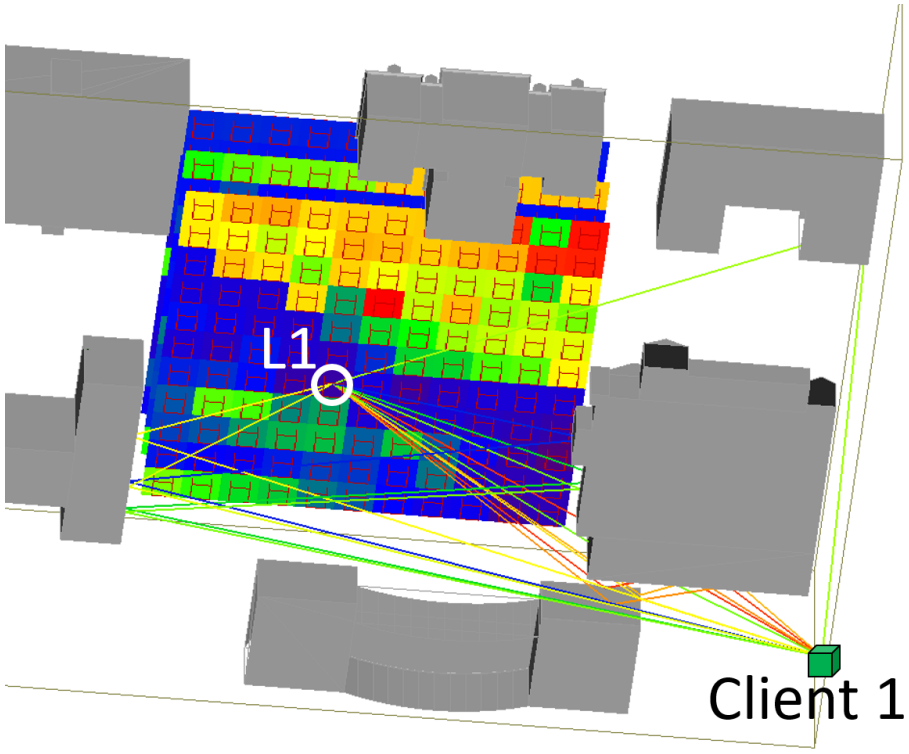

Figure 5.6: 3D SNR heatmap

Figure 5.7 shows the comparison between measured SNR and ray tracing predicted SNR. While ray tracing simulations can model the reality well (Figure 5.7(a)), it can also be less accurate for some clients (Figure 5.7(b)).

The simulations can be offloaded to cloud. Real time results [175, 176] are achievable through parallelization and tradeoffs with accuracy. In this project, however, we conduct simulations offline.

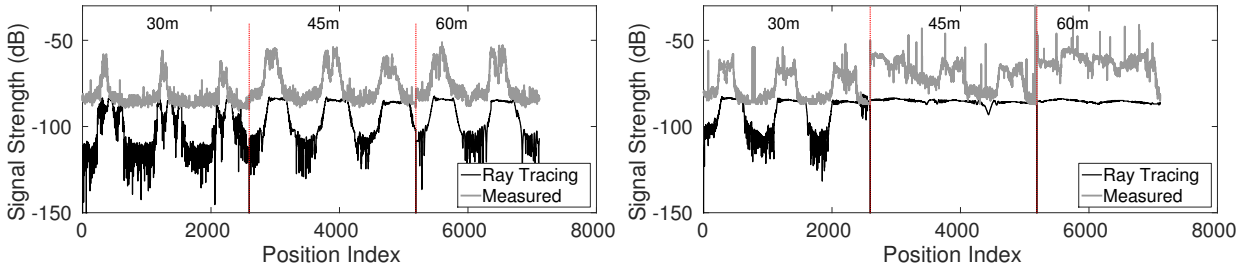

Figure 5.7: (a) Ray tracing can capture variation in SNR well. (b) Ray tracing can be inaccurate in some cases

5.3.2 Fusion of ray tracing and measurements

While ray-tracing predictions are promising, hovering a drone entirely based on predictions gave poor performance. We define *SNR gain* of a candidate position for drone hovering as the difference between the measured SNR of that position and the median of measured SNRs of all candidate positions (more details in Section 5.4). We take top 10%-ile high SNR drone positions from ray-tracing predictions and plot a CDF of their *SNR gain* in Figure 5.8. Evidently, the median is close to 0 suggesting that ray-tracing alone is not enough. The low accuracy arises from inexact terrain modeling, imprecise material absorptions, unmodeled foliage and limited number of rays/reflections/diffractions used to reduce complexity. *Oracle* is an imaginary system which can magically predict measurements to 100% accuracy. We observe in Figure 5.8 that Oracle has a substantial *SNR gain*. *DroneNet* achieves 57% of these gains by fusing ray-tracing predictions with sparse measurements, as elaborated below.

1. Perform ray tracing simulations to obtain sum-throughput heatmap as outlined in Section 5.3.1
2. Select the top 10%ile drone locations in the sum-throughput heatmap. Let this set of points be denoted as ht
3. Divide the entire region into chunks of size $25m^2$. Find the chunk that contains the largest number of positions from ht.
4. Scan the chunk determined from the above step to conduct physical throughput measure-

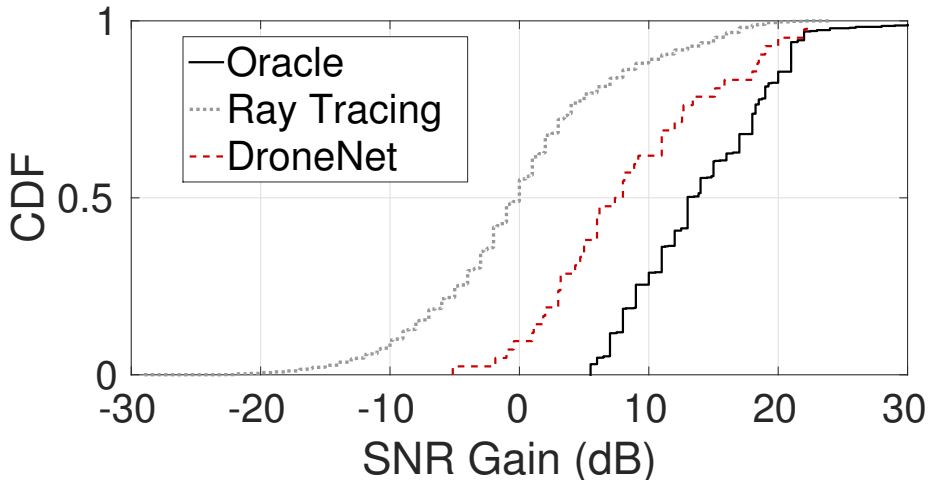

Figure 5.8: Ray tracing predictions fail to capture SNR gains

ments. The position with maximum throughput in the chunk based on measurements is selected for hovering.

After determining the optimal location, the drone needs to relocate there. The relocation accuracy allows some tolerance because there will be many spatially contiguous points with similar throughput around the optimal position.

5.4 Evaluation

5.4.1 Experimental Setup

Our experimental platform consists of an X8 quadcopter from 3D Robotics [177]. We placed an Almond WiFi AP [178], powered by a LiPo battery, on the quadcopter and set it to transmit at a frequency of 2.437GHz, (20MHz bandwidth), with a transmit power of 28dBm. Whereas we use WiFi for the client-drone link, it can also be a cellular link with a femtocell (instead of the WiFi AP) operating in a designated LTE band. An Android phone was additionally used to time-stamp and location stamp the packets sent out by the Almond AP. 7 clients running linux on Raspberry Pis were spread in an area ($160 \times 280m$) around the Engineering Quad at the University of Illinois, as shown in Figure 5.9(a).

The Raspberry Pis were connected to an Atheros WiFi dongle. The AP transmitted 400 packets per second, which were captured by the clients from which SNR was computed. During the packet transmissions, the drone flew in a raster scan at a speed of 1m/s covering the area ($50 \times 68m$) highlighted in Figure 5.9(a). Three different heights—30, 45 and 60 m—were used for drone flights above this area. This provides us a platform to measure the spatial heatmap of SNR variation.[2]

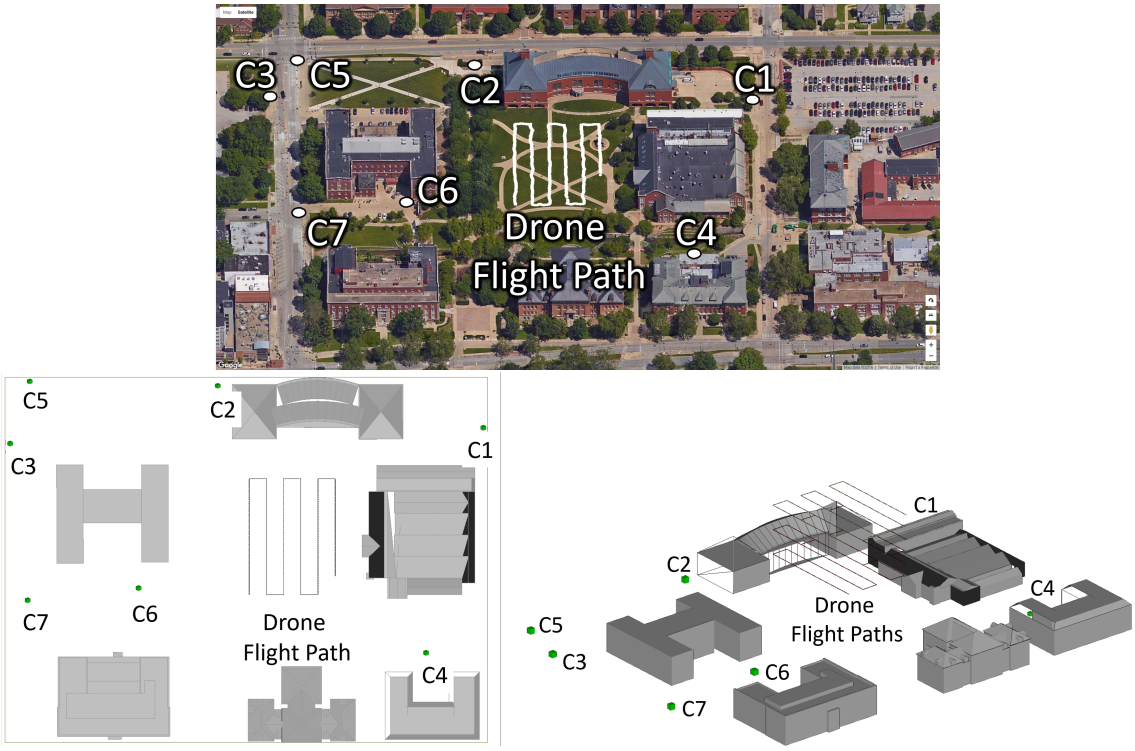

Figure 5.9: (a) Client locations and drone flight path (b) 3D Terrain Model – top view (c) 3D Model – perspective view

The 3D terrain model of most buildings around the Quad were obtained from Google 3D Warehouse [172]. Some of the unavailable models were manually created using Auto-CAD softwares by borrowing relevant design diagrams from the architecture department. The top and perspective view of the buildings from Figure 5.9(a) is modeled in Figure 5.9 (b, c). The model includes 3D terrain structures, client locations and drone flight path. The 3D model was

[2]Experiments were conducted as per FAA regulations. Operator was trained by a general aviation pilot and flight permissions were obtained from campus police.

input to Remcom Wireless Insite [169], producing another spatial heatmap of SNR variations used for predictions.

Let the *Oracle* be a system whose ray-tracing predictions match exactly with measurements. We define *Random* as a system where the drone randomly chooses a position to hover. We define *Ray Tracing* as a system where the drone hovers at the best location predicted by ray tracing simulations. We characterize performance gains of systems – *Ray Tracing*, *iMob*, and *Oracle* over *Random*. We use throughput as the performance metric and translate SNR into throughput using Shannon's equation.

5.4.2 Performance Results

Single client throughput: Figure 5.10 quantifies the gain in throughput. *iMob* gain varies from $1.2x$ to $4.2x$ over various clients. While *RayTrace* gains has a significant difference from *Oracle* gains, *iMob* captures reasonable *Oracle* gains.

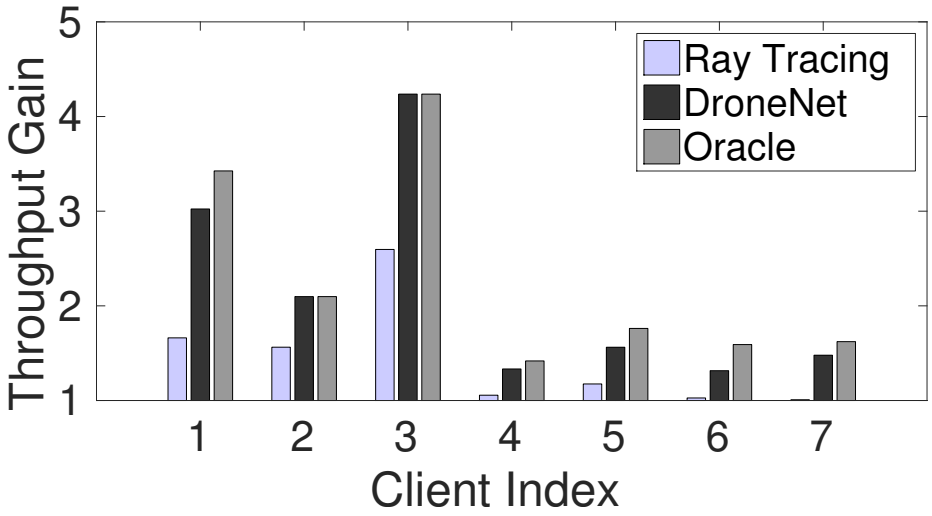

Figure 5.10: *iMob* gains are comparable with the *Oracle*

Multi client throughput: Figure 5.11 shows the pattern of gain across multiple gains. While

the gains decrease with higher clients, we need not optimize for all clients simultaneously. Power law [179] indicates that 50% traffic demand comes from 1% of the users who from the bottleneck. Addressing the bottleneck will benefit all other users.

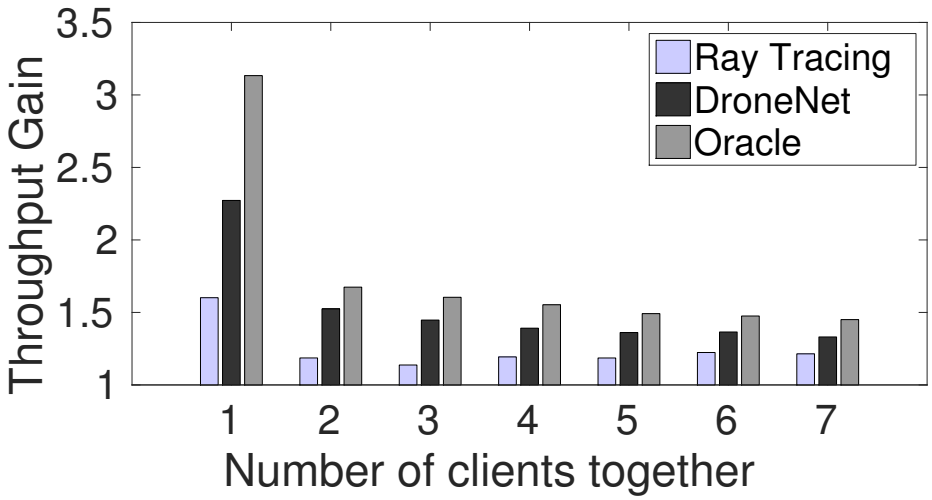

Figure 5.11: *iMob* provides up to 44% gains for 3 clients and 36% for 5 clients

Gains with constrained motion: Figure 5.12 computes the gains over various spatial constraints of drone motion. The experimental area, A, is partitioned into 4, 6, and 8 regions shown as $A/4$, $A/6$ and $A/8$ on the x-axis. Even with constrained motion, there is a graceful degradation in gains, suggesting that enough diversity from shadowing exists. This also suggests that even constrained mobility can offer gains. Figure 5.13 zooms in to show the CDF of gains for $A/6$. On an average, *iMob* achieves 57% of oracle gains with only 10% of measurement overhead.

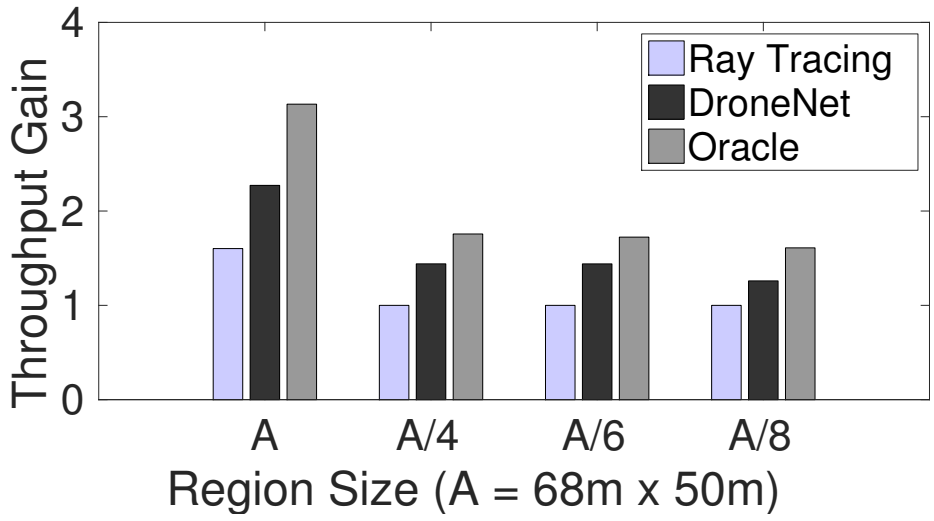

Figure 5.12: 25 ~ 45% gains, with constrained mobility

5.5 Related Work

5.5.1 Mobility

Beamforming: While Beamforming techniques [180] help addressing similar problems as *DroneNet* by steering the radiation pattern based on demand, we believe combining beamforming with physical mobility can offer higher flexibility.

Mobile Infrastructure: *DroneNet*'s notion of mobile infrastructure is similar to Google Loon [181] and Facebook drones [182]. *DroneNet*, however, attempts to increase spectrum efficiency and address dynamic demands even in high density urban networks and not just in remote areas. DARPA Landroids [183] use autonomous robots for offering communication in challenging terrains for military. We believe that ideas in *DroneNet* will be useful for Landroids. Cellular towers are moved on vans [184] during sporting events. While useful, it viable only during large gatherings. A very early version of our work was presented as a Mobicom poster [185].

In indoor spaces, *iMob* [186, 187] shows that Roomba robots can enhance throughput by

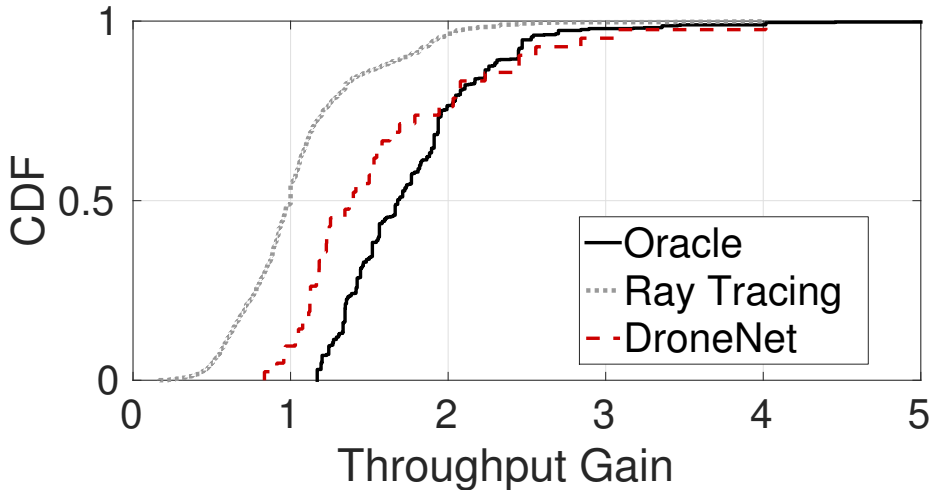

Figure 5.13: CDF of gains for A/6 region size

exploiting multipath opportunities. *DroneNet* operates outdoors and whereas *iMob* directly uses empirical measurements for finding the optimal AP position, for a large outdoor 3D search space, such measurements become costly. *DroneNet* uses ray-tracing to reduce this cost.

A swarm of drones is used in [188] to demonstrate multi-hop connectivity. In contrast, we focus on predicting optimal placement of the drones utilizing ray-tracing techniques.

5.5.2 Spatial SNR Modeling

Path loss Models: Path loss models characterize RF attenuation as a power of propagation distance. The exponent and variation properties are determined empirically and used for predictions [189–191]. However, its accuracy is limited and doesn't capture sudden SNR changes at building boundaries.

Ray Tracing: Ray tracing has been studied extensively in the context of cellular network planning. Works in [192, 193] show that ray tracing models can predict the SNR in urban and semi-urban areas. Works in [194, 195] attempt to decrease the complexity for Manhattan style urban areas by transforming 3d ray tracing into 2d ray tracing. Similarly [196] proposes an an-

gular partitioning technique to decrease complexity. *DroneNet* benefits from these techniques and, moreover, incorporates selective measurements for refining the model.

5.6 Discussion and On-going work

Client Mobility: We do not address mobile clients in this project. Plethora of research exists in vehicular networking community for exploiting the predictive nature of mobility [197]. We believe it would be an interesting approach to combine ray tracing with mobility models to address the challenges. *DroneNet*'s ray-tracing models can be generalized to a path instead of a point, however, we leave this for future work.

OFDM/MIMO: SNR is not a good indicator of throughput for multi-carrier OFDM systems. While direct throughput assessment (with Iperf) was infeasible because of flying constraints, it is possible to extend the ray-tracing simulations to OFDM by observing the channel frequency response (CFR). Evaluation can be conducted with platforms that export CFR [144]. Perhaps, more opportunities at the subcarrier/antenna level from OFDM/MIMO can be used for optimizations. We treat them as separate problems and leave them for future work.

Location Requirement: *DroneNet*'s ray-tracing module requires client locations to be known with an accuracy of couple of meters. Since we attempt to mitigate large shadows caused by terrain profile, we do not need wavelength level accuracy. GPS location sharing may have privacy concerns. We do not address these concerns in this project, our focus is to explore the possibilities.

5.7 Conclusion

Demand for cellular traffic continues to increase while the spectrum resources are limited. Cellular users also suffer from intermittent regions of poor connectivity due to wireless shadows and dead zones caused by buildings. We envision a system of drones that can extend cell towers and mitigate the dynamic connectivity issues while also facilitating efficient use of the scarce spectrum. Our system *DroneNet* explores a key problem of determining optimal drone placements. While we scratch the surface, much more remains to be done. Early prototype with real flights offers sufficient promise for a long term research engagement.

Chapter 6

Conclusion

In this thesis, we develop mobility inferencing techniques for emerging IoT applications. The core contribution is in the fusion of multi-modal sensory data from wireless signals and inertial sensors with application specific models to infer mobility. We specifically discussed three diverse class of applications that leverage from this principle.

Drone Reliability: Chapter 2 discusses GPS based drone orientation tracking. Drones rely on inertial sensors for navigation and control. Unfortunately, the unreliability of inertial sensors is the key reason for drone crashes today. In chapter 2, we proposed to augment drones with GPS based orientation tracking, and hence replace inertial sensors under failure. We placed multiple GPS receivers on arms of a drone to translate their relative position estimates into drone orientation. This was a challenging problem because a small sized drone entails cm level relative position accuracy to achieve high orientation accuracy. Moreover, aggressive maneuvers of a small drone can introduce sudden SNR changes and loss of locks with GPS satellites, thus exacerbating the problem. Fortunately, the small size of the drone together with geometric model of GPS receiver placement, allowed an opportunity to use an advanced particle filter based design to tackle the problem. *SafetyNet* results demonstrate high accuracy together with robustness across a diverse setting.

Sports Analytics: As discussed in chapter 3, tracking sports objects using embedded sensors poses completely new challenges. A cricket or baseball can move at 100 miles per hour and spin at 40 revolutions per second. None of the existing wireless or inertial localization techniques can handle such mobility regimes. Similarly, a freely falling object precludes measurement

of the gravity vector thereby eliminating an important opportunity for rotation estimation. Finally, sensor instrumentation has to be sparse to be less disruptive to the game. While, these are novel challenges posed by the application, we showed how unique opportunities open up. Unlike smartphone motion, a ball motion follows well defined laws of physics. Combining the sparse and noisy sensor data of limited degrees of freedom with physics models of ball motion enabled tracking of the location and spin with promising accuracy.

Robotic Wireless Networks: In chapters 4 and 5, we explored mobile APs and base-stations that can physically move on robots and drones to enhance the performance of a network. While the key challenge is to quickly find the optimal position for the mobile AP, *iMob* and *DroneNet* combine sensed wireless data with wireless channel models. Specifically, in *iMob* we exploited the statistical variation principles of wireless indoor multipath, whereas in *DroneNet*, we combined measured channel strengths with environment aware 3D propagation models to considerably reduce the search space for positioning.

Future Work: We believe that the design principle of "sensor fusion with application specific models" can be employed towards solving numerous emerging problems in the IoT space. One such application space immediately under our radar is in smart and connected health care. For example, tracking body motion consisting of movements of upper, lower limbs, fingers, head, chest motion and skin vibration provides lenses into various physiological phenomena. While medical diagnosis already benefits from careful analysis of motion, it must be performed at clinics or special facilities. With wearables devices like smartphones and watches becoming popular, combined with advances in tracking and machine learning algorithms, new opportunities are on the horizon. For instance, it should be now possible to continuously analyze a patient's gait patterns from embedded motion sensors, ultimately enabling "fall detection" in elderly people. Similarly, respiratory, motor, and cardiovascular disorders are other examples of applications that can leverage from motion analytics.

While the problem is challenging because of complex nature of body motion, we observe that there are rich biological, anatomical and physiological models that dictate body motion. Fusion of inertial, wireless, and biological data with these models can provide a valuable opportunity for tracking. We look forward to exploring this space in the future.

References

[1] "Amazon echo." https://www.amazon.com/echo.

[2] "Google glass." http://www.x.company/glass/.

[3] "Fitbit." https://www.fitbit.com/home.

[4] "Oculus rift." https://www.oculus.com/rift/.

[5] "Samsung gear s3." https://www.samsung.com/us/explore/gear-s3/?cid=ppc-.

[6] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *Proceedings of the 18th annual international conference on Mobile computing and networking – MOBICOM 2012*, pp. 293–304, ACM, 2012.

[7] P. Zhou, M. Li, and G. Shen, "Use it free: Instantly knowing your phone attitude," in *Proceedings of the 20th annual international conference on Mobile computing and networking – MOBICOM 2014*, pp. 605–616, ACM, 2014.

[8] J. L. Crassidis, F. L. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods," *Journal of guidance, control, and dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[9] C. E. Cohen, E. G. Lightsey, B. W. Parkinson, and W. A. Feess, "Space flight tests of attitude determination using gps," *International Journal of Satellite Communications*, 1994.

[10] G. Lachapelle, M. Cannon, G. Lu, and B. Loncarevic, "Shipborne gps attitude determination during mmst-93," *Oceanic Engineering, IEEE Journal of*, 1996.

[11] R. J. Moore, S. Thurrowgood, D. Bland, D. Soccol, and M. V. Srinivasan, "A stereo vision system for uav guidance," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 3386–3391, IEEE, 2009.

[12] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M. E. Munich, "The vslam algorithm for robust localization and mapping," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 24–29, IEEE, 2005.

[13] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, pp. 775–784, Ieee, 2000.

[14] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pp. 205–218, ACM, 2005.

[15] J. Xiong and K. Jamieson, "Arraytrack: a fine-grained indoor location system," in *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pp. 71–84, 2013.

[16] S. Kumar, S. Gil, D. Katabi, and D. Rus, "Accurate indoor localization with zero start-up cost," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, pp. 483–494, ACM, 2014.

[17] G. M. Siouris, G. Chen, and J. Wang, "Tracking an incoming ballistic missile using an extended interval kalman filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 1, pp. 232–240, 1997.

[18] G. Peskir and A. Shiryaev, *Optimal stopping and free-boundary problems*. Springer, 2006.

[19] T. S. Ferguson, "Who solved the secretary problem?," *Statistical science*, pp. 282–289, 1989.

[20] Y. Zhang, A. Chamseddine, C. Rabbath, B. Gordon, C.-Y. Su, S. Rakheja, C. Fulford, J. Apkarian, and P. Gosselin, "Development of advanced fdd and ftc techniques with application to an unmanned quadrotor helicopter testbed," *Journal of the Franklin Institute*, vol. 350, no. 9, pp. 2396–2422, 2013.

[21] "diydrones." `http://diydrones.com/forum/topics/problem-quad-copter-spins-around-itseli`

[22] "Google forum." `https://groups.google.com/forum/#!msg/drones-discuss/fnWYM48pGys/kwPI_q_qv0QJ`.

[23] "Drone crash survey." `https://docs.google.com/forms/d/1R4MaX8iZWRoyzKJ6rKr0U-p3x-6j0ZVm5a0p83hHty0/viewanalytics`.

[24] G. De Pasquale and A. Somà, "Reliability testing procedure for mems imus applied to vibrating environments," *Sensors*, vol. 10, no. 1, pp. 456–474, 2010.

[25] "3d robotics: Pixhawk." `https://3dr.com/pixhawk-is-ready-for-takeoff/`.

[26] "Dji phantom 4." `https://www.dji.com/product/phantom-4/info`.

[27] X. Liu and R. Randall, "Blind source separation of internal combustion engine piston slap from other measured vibration signals," *Mechanical Systems and Signal Processing*, vol. 19, no. 6, pp. 1196–1208, 2005.

[28] M. Wells, *Attenuating magnetic interference in a UAV system*. PhD thesis, Carleton University Ottawa, 2008.

[29] V. Malyavej, P. Torteeka, S. Wongkharn, and T. Wiangtong, "Pose estimation of unmanned ground vehicle based on dead-reckoning/gps sensor fusion by unscented kalman filter," in *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*, vol. 1, pp. 395–398, IEEE, 2009.

[30] Y. S. Suh, "Attitude estimation by multiple-mode kalman filters," *IEEE Transactions on industrial electronics*, vol. 53, no. 4, pp. 1386–1389, 2006.

[31] Y.-C. Lai and S.-S. Jan, "Attitude estimation based on fusion of gyroscopes and single antenna gps for small uavs under the influence of vibration," *GPS solutions*, vol. 15, no. 1, pp. 67–77, 2011.

[32] W. Hedgecock *et al.*, "Regtrack: a differential relative gps tracking solution," in *Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services – MOBISYS 2013*, 2013.

[33] W. Hedgecock *et al.*, "Accurate real-time relative localization using single-frequency gps," in *SENSYS 2014*, ACM, 2014.

[34] A. R. Conway, *Autonomous control of an unstable model helicopter using carrier phase GPS only*. PhD thesis, stanford university, 1995.

[35] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: Accurate, energy-aware road traffic delay estimation using mobile phones," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, (New York, NY, USA), pp. 85–98, ACM, 2009.

[36] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive gps-based positioning for smartphones," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, (New York, NY, USA), pp. 299–314, ACM, 2010.

[37] "Lipo 4s battery." `http://www.droneshop.biz/4s-10000mah-lipo-low-weight-high-efficiency.html`.

[38] "Dji matrice 600." `http://www.dji.com/product/matrice600`.

[39] "Magnetic interference causing crashes." `http://diydrones.com/forum/topics/crazy-magnetic-interference`.

[40] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements and Performance Second Edition*. Lincoln, MA: Ganga-Jamuna Press, 2006.

[41] R. B. Langley, "Glonass: review and update," *GPS world*, vol. 8, no. 7, pp. 46–51, 1997.

[42] "Galileo." `https://en.wikipedia.org/wiki/Galileo_(satellite_navigation)`.

[43] B. Zhang, P. J. Teunissen, and D. Odijk, "A novel un-differenced ppp-rtk concept," *Journal of Navigation*, vol. 64, no. S1, pp. S180–S191, 2011.

[44] D. Laurichesse, F. Mercier, J.-P. BERTHIAS, P. Broca, and L. Cerri, "Integer ambiguity resolution on undifferenced gps phase measurements and its application to ppp and satellite precise orbit determination," *Navigation*, vol. 56, no. 2, pp. 135–149, 2009.

[45] E. Kaplan and C. Hegarty, *Understanding GPS: principles and applications*. Artech house, 2005.

[46] B. W. Parkinson and P. K. Enge, "Differential gps," *Global Positioning System: Theory and applications.*, vol. 2, pp. 3–50, 1996.

[47] C. Mekik and M. Arslanoglu, "Investigation on accuracies of real time kinematic gps for gis applications," *Remote Sensing*, vol. 1, no. 1, pp. 22–35, 2009.

[48] G. Blewitt, "Basics of the gps technique: observation equations," *Geodetic applications of GPS*, pp. 10–54, 1997.

[49] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," *Matrix*, vol. 58, no. 15-16, pp. 1–35, 2006.

[50] "Rotation vectors." `http://farside.ph.utexas.edu/teaching/301/lectures/node100.html`.

[51] "Vector cross product." `http://soe.rutgers.edu/~meer/GRAD561/ADD/antisymm.pdf`.

[52] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[53] C. E. Cohen and B. W. Parkinson, "Integer ambiguity resolution of the gps carrier for spacecraft attitude determination," in *Guidance and Control 1992*, vol. 1, pp. 107–118, 1992.

[54] A. Conway, P. Montgomery, S. Rock, R. Cannon, and B. Parkinson, "A new motion-based algorithm for gps attitude integer resolution," *Navigation*, vol. 43, no. 2, pp. 179–190, 1996.

[55] P. De Jonge and C. Tiberius, "The lambda method for integer ambiguity estimation: implementation aspects," *Publications of the Delft Computing Centre, LGR-Series*, vol. 12, no. 12, pp. 1–47, 1996.

[56] E. Sutton and R. Collins, "Calibration of differential phase map compensation using single axis rotation," in *PROCEEDINGS OF ION GPS*, vol. 11, pp. 1831–1842, INSTITUTE OF NAVIGATION, 1998.

[57] J. Vallet, F. Panissod, C. Strecha, and M. Tracol, "Photogrammetric performance of an ultra light weight swinglet uav," in *UAV-g*, no. EPFL-CONF-169252, 2011.

[58] J. O. Street, R. J. Carroll, and D. Ruppert, "A note on computing robust regression estimates via iteratively reweighted least squares," *The American Statistician*, vol. 42, no. 2, pp. 152–154, 1988.

[59] M. E. Pittelkau, "Rotation vector in attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 6, pp. 855–860, 2003.

[60] S. Harwin and A. Lucieer, "Assessing the accuracy of georeferenced point clouds produced via multi-view stereopsis from unmanned aerial vehicle (uav) imagery," *Remote Sensing*, vol. 4, no. 6, pp. 1573–1599, 2012.

[61] "Wide area augmentation system." `https://en.wikipedia.org/wiki/Wide_Area_Augmentation_System`.

[62] Y. Li, K. Zhang, C. Roberts, and M. Murata, "On-the-fly gps-based attitude determination using single-and double-differenced carrier phase measurements," *GPS Solutions*, vol. 8, no. 2, pp. 93–102, 2004.

[63] D. E. Kirkpatrick, "Design of a hardware platform for gps-based orientation sensing," 2015.

[64] A. G. Evans, "Roll, pitch, and yaw determination using a global positioning system receiver and an antenna periodically moving in a plane," *Marine Geodesy*, vol. 10, no. 1, pp. 43–52, 1986.

[65] M. Ueno, R. Santerre, and A. Kleusberg, "Direct determination of angular velocity using gps," *Journal of Navigation*, vol. 53, no. 02, pp. 371–379, 2000.

[66] P. Montgomery, H. Uematsu, and B. Parkinson, "Analysis of angular velocity determination using gps," in *ION GPS-94*, pp. 697–706, 1994.

[67] J. Wang, M. P. Stewart, and M. Tsakiri, "Modelling glonass measurements for precise positioning," *Survey Review*, vol. 36, no. 280, pp. 110–120, 2001.

[68] S. Han, L. Dai, and C. Rizos, "A new data processing strategy for combined gps/glonass carrier phase-based positioning," in *Proc. ION GPS-99*, pp. 1619–1627, 1999.

[69] T. Tsujii, M. Harigae, T. Inagaki, and T. Kanai, "Flight tests of gps/glonass precise positioning versus dual frequency kgps profile," *Earth, planets and space*, vol. 52, no. 10, pp. 825–829, 2000.

[70] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang, "Walkie-markie: indoor pathway mapping made easy," in *Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation – NSDI 2013*, pp. 85–98, USENIX Association, 2013.

[71] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to wardrive: unsupervised indoor localization," in *Proceedings of the 10th international conference on Mobile systems, applications, and services – MOBISYS 2012*, pp. 197–210, ACM, 2012.

[72] J. G. Manweiler, P. Jain, and R. Roy Choudhury, "Satellites in our pockets: an object positioning system using smartphones," in *Proceedings of the 10th international conference on Mobile systems, applications, and services – MOBISYS 2012*, pp. 211–224, ACM, 2012.

[73] A. T. Mariakakis, S. Sen, J. Lee, and K.-H. Kim, "Sail: Single access point-based indoor localization," in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services – MOBISYS 2014*, MobiSys '14, (New York, NY, USA), pp. 315–328, ACM, 2014.

[74] D. B. Kingston and R. W. Beard, "Real-time attitude and position estimation for small uavs using low-cost sensors," in *AIAA 3rd unmanned unlimited technical conference, Workshop and exhibit*, pp. 2004–6488, sn, 2004.

[75] W. Y. Liang, W. T. Miao, L. J. Hong, X. C. Lei, and Z. Chen, "Attitude estimation for small helicopter using extended kalman filter," in *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, pp. 577–581, IEEE, 2008.

[76] D. Marina, H. Garcia, F. J. Pereda, J. M. Giron-Sierra, and F. Espinosa, "Uav attitude estimation using unscented kalman filter and triad," *Industrial Electronics, IEEE Transactions on*, vol. 59, no. 11, pp. 4465–4474, 2012.

[77] J. L. Crassidis and F. L. Markley, "Unscented filtering for spacecraft attitude estimation," *Journal of guidance, control, and dynamics*, vol. 26, no. 4, pp. 536–542, 2003.

[78] G. G. Rigatos, "Nonlinear kalman filters and particle filters for integrated navigation of unmanned aerial vehicles," *Robotics and Autonomous Systems*, vol. 60, no. 7, pp. 978–995, 2012.

[79] R. Van Der Merwe and E. A. Wan, "Sigma-point kalman filters for integrated navigation," in *Proceedings of the 60th Annual Meeting of the Institute of Navigation (ION)*, pp. 641–654, 2004.

[80] S. Thurrowgood, D. Soccol, R. J. Moore, D. Bland, and M. V. Srinivasan, "A vision based system for attitude estimation of uavs," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 5725–5730, IEEE, 2009.

[81] C. Demonceaux, P. Vasseur, and C. Pégard, "Omnidirectional vision on uav for attitude computation," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 2842–2847, IEEE, 2006.

[82] G. L. Barrows, J. S. Chahl, and Y. V. Srinivasan, "Biomimetic visual sensing and flight control," in *Proc. Bristol UAV Conf*, pp. 159–168, 2002.

[83] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 15–22, IEEE, 2014.

[84] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *Computer Vision–ECCV 2014*, pp. 834–849, Springer, 2014.

[85] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pp. 225–234, IEEE, 2007.

[86] Nike, "Footwear having sensor system." Patent US 8676541 B2.

[87] B. Zhou, H. Koerger, M. Wirth, C. Zwick, C. Martindale, H. Cruz, B. Eskofier, and P. Lukowicz, "Smart soccer shoe: monitoring foot-ball interaction with shoe integrated textile pressure sensor matrix," in *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, pp. 64–71, ACM, 2016.

[88] "Firstv1sion." `http://www.firstv1sion.com/en/product/`.

[89] "Goal-line technology." `https://en.wikipedia.org/wiki/Goal-line_technology`.

[90] `http://www.jamesgibbard.co.uk/electronics/bluetoothcontrolledledflashingfrisbee`.

[91] "High range gyroscopes." `http://www.analog.com/en/products/mems/gyroscopes.html`.

[92] "D2m." `http://d2m-inc.com/`.

[93] "High speed camera." `http://www.untamedscience.com/filmmaking/advanced-filmmaking/high-speed-video-slow-motion/`.

[94] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu, "Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, pp. 237–248, ACM, 2014.

[95] R. B. Langley, "Dilution of precision," *GPS world*, vol. 10, no. 5, pp. 52–59, 1999.

[96] E. Swanson, "Geometric dilution of precision," *Navigation*, vol. 25, no. 4, pp. 425–429, 1978.

[97] D. Hahnel, W. Burgard, D. Fox, and S. Thrun, "An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 1, pp. 206–211, IEEE, 2003.

[98] S. Gupta, D. Morris, S. Patel, and D. Tan, "Soundwave: using the doppler effect to sense gestures," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1911–1914, ACM, 2012.

[99] "Decawave."

[100] P. Zhou, M. Li, and G. Shen, "Use it free: instantly knowing your phone attitude," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, pp. 605–616, ACM, 2014.

[101] N.-C. Tsai and C.-Y. Sue, "Stability and resonance of micro-machined gyroscope under nonlinearity effects," *Nonlinear Dynamics*, vol. 56, no. 4, pp. 369–379, 2009.

[102] R. Hach, "Symmetric double sided two-way ranging," *IEEE P802*, vol. 15, pp. 802–15, 2005.

[103] C. Baker, "A calculation of cricket ball trajectories," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 224, no. 9, pp. 1947–1958, 2010.

[104] "adidas micoach smart ball." `http://www.adidas.com/us/micoach-smart-ball/G83963.html`.

[105] F. K. Fuss, R. M. Smith, and A. Subic, "Determination of spin rate and axes with an instrumented cricket ball," *Procedia Engineering*, vol. 34, pp. 128–133, 2012.

[106] F. K. Fuss, N. Lythgo, R. M. Smith, A. C. Benson, and B. Gordon, "Identification of key performance parameters during off-spin bowling with a smart cricket ball," *Sports Technology*, vol. 4, no. 3-4, pp. 159–163, 2011.

[107] F. K. Fuss and R. M. Smith, "Accuracy performance parameters of seam bowling, measured with a smart cricket ball," *Procedia Engineering*, vol. 72, pp. 435–440, 2014.

[108] R. S. McGinnis and N. C. Perkins, "A highly miniaturized, wireless inertial measurement unit for characterizing the dynamics of pitched baseballs and softballs," *Sensors*, vol. 12, no. 9, pp. 11933–11945, 2012.

[109] F. Fuss, R. Ferdinands, B. Doljin, and A. Beach, "Development of a smart cricket ball and advanced performance analysis of spin bowling," in *ICSST 2014: Advanced Technologies in Modern Day Sports*, pp. 588–595, Institute for Sports Research (ISR), 2014.

[110] K. King, N. C. Perkins, H. Churchill, R. McGinnis, R. Doss, and R. Hickland, "Bowling ball dynamics revealed by miniature wireless mems inertial measurement unit," *Sports Engineering*, vol. 13, no. 2, pp. 95–104, 2011.

[111] "Hawk-eye." `https://en.wikipedia.org/wiki/Hawk-Eye/`.

[112] "Hot spot." `https://en.wikipedia.org/wiki/Hot_Spot_(cricket)/`.

[113] P. Halvorsen, S. Sægrov, A. Mortensen, D. K. Kristensen, A. Eichhorn, M. Stenhaug, S. Dahl, H. K. Stensland, V. R. Gaddam, C. Griwodz, *et al.*, "Bagadus: an integrated system for arena sports analytics: a soccer case study," in *Proceedings of the 4th ACM Multimedia Systems Conference*, pp. 48–59, ACM, 2013.

[114] Y. Seo, S. Choi, H. Kim, and K.-S. Hong, "Where are the ball and players? soccer game analysis with color-based tracking and image mosaick," in *International Conference on Image Analysis and Processing*, pp. 196–203, Springer, 1997.

[115] X. Yu, C. Xu, H. W. Leong, Q. Tian, Q. Tang, and K. W. Wan, "Trajectory-based ball detection and tracking with applications to semantic analysis of broadcast soccer video," in *Proceedings of the eleventh ACM international conference on Multimedia*, pp. 11–20, ACM, 2003.

[116] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, pp. 173–184, ACM, 2010.

[117] D. Niculescu and B. Nath, "Ad hoc positioning system (aps) using aoa," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, pp. 1734–1743, Ieee, 2003.

[118] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to wardrive: unsupervised indoor localization," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 197–210, ACM, 2012.

[119] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *Proceedings of the 18th annual international conference on Mobile computing and networking*, pp. 293–304, ACM, 2012.

[120] D. Vasisht, S. Kumar, and D. Katabi, "Decimeter-level localization with a single wifi access point," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pp. 165–178, 2016.

[121] C. McElroy, D. Neirynck, and M. McLaughlin, "Comparison of wireless clock synchronization algorithms for indoor location systems," in *2014 IEEE International Conference on Communications Workshops (ICC)*, pp. 157–162, IEEE, 2014.

[122] S. Madgwick, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," *Report x-io and University of Bristol (UK)*, 2010.

[123] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[124] W. Y. Liang, W. T. Miao, L. J. Hong, X. C. Lei, and Z. Chen, "Attitude estimation for small helicopter using extended kalman filter," in *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, pp. 577–581, IEEE, 2008.

[125] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[126] J. M. Pflimlin, T. Hamel, and P. Souères, "Nonlinear attitude and gyroscope's bias estimation for a vtol uav," *International Journal of Systems Science*, vol. 38, no. 3, pp. 197–210, 2007.

[127] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill, "Designing high performance enterprise wi-fi networks.," in *NSDI*, 2008.

[128] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, 2009.

[129] G. A. Lewis, S. Echeverría, S. Simanta, B. Bradshaw, and J. Root, "Cloudlet-based cyber-foraging for mobile systems in resource-constrained edge environments," in *Companion Proceedings of the 36th International Conference on Software Engineering*, pp. 412–415, ACM, 2014.

[130] M. Jain, J. I. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha, "Practical, real-time, full duplex wireless," in *ACM Mobicom*, 2011.

[131] H. V. Balan, R. Rogalin, A. Michaloliakos, K. Psounis, and G. Caire, "Achieving high data rates in a distributed mimo system," in *ACM Mobicom*, 2012.

[132] F. Adib, S. Kumar, O. Aryan, S. Gollakota, and D. Katabi, "Interference alignment by motion," in *ACM MOBICOM*, 2013.

[133] "irobot roomba." http://www.irobot.com/us/learn/home/roomba.aspx.

[134] "Quadcopters." `http://www.ted.com/talks/raffaello_d_andrea_the_astounding_athletic_power_of_quadcopters.html`.

[135] R. R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: The next computing revolution," in *Proceedings of the 47th Design Automation Conference*, DAC '10, ACM, 2010.

[136] "Amazon prime air." http://www.amazon.com/b?node=8037720011.

[137] "Robots are becoming ready to work among us." http://www.technologyreview.com/view/522646/robots-are-becoming-ready-to-work-among-us/.

[138] "Smart robot car chassis kit for arduino (works with official arduino boards)." http://www.dx.com/p/smart-robot-car-chassis-kit-for-arduino-transparent-yellow-152984.

[139] "Advance in robotic technology." http://www.everything-robotic.com/.

[140] J. B. Andersen, T. S. Rappaport, and S. Yoshida, "Propagation measurements and models for wireless communications channels," *Communications Magazine, IEEE*, vol. 33, no. 1, pp. 42–49, 1995.

[141] P. Shankar, "Outage probabilities of a mimo scheme in shadowed fading channels with micro-and macrodiversity reception," *Wireless Communications, IEEE Transactions on*, vol. 7, no. 6, pp. 2015–2019, 2008.

[142] M. Gowda, N. Roy, and R. Roy Choudhury, "Infrastructure mobility: A what-if analysis," *Hotnets*, 2014.

[143] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: Gathering 802.11n traces with channel state information," *SIGCOMM Comput. Commun. Rev.*, vol. 41, pp. 53–53, Jan. 2011.

[144] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," *ACM SIGCOMM*, 2010.

[145] S. Sen, R. R. Choudhury, B. RadunoviÄĞ, and T. Minka, "You are facing the mona lisa: spot localization using phy layer information," in *ACM Mobisys*, 2012.

[146] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: extracting a secret key from an unauthenticated wireless channel," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pp. 128–139, ACM, 2008.

[147] J. Zhang, M. H. Firooz, N. Patwari, and S. K. Kasera, "Advancing wireless link signatures for location distinction," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, (New York, NY, USA), pp. 26–37, ACM, 2008.

[148] "Google project loon." `http://www.google.com/loon/`.

[149] "Landroid robots to support communications in urban combat." http://defense-update.com/products/l/landroids.htm.

[150] D. Gesbert, M. Shafi, D.-s. Shiu, P. J. Smith, and A. Naguib, "From theory to practice: an overview of mimo space-time coded wireless systems," *JSAC*, vol. 21, no. 3, pp. 281–302, 2003.

[151] J. Litva and T. K. Lo, *Digital beamforming in wireless communications*. Artech House, Inc., 1996.

[152] A. Miu, G. Tan, H. Balakrishnan, and J. Apostolopoulos, "Divert: fine-grained path selection for wireless lans," in *ACM Mobisys*, 2004.

[153] A. Balasubramanian, R. Mahajan, A. Venkataramani, B. N. Levine, and J. Zahorjan, "Interactive wifi connectivity for moving vehicles," in *ACM SIGCOMM*, 2008.

[154] M. Zavlanos, A. Ribeiro, and G. Pappas, "Network integrity in mobile robotic networks," in *IEEE Transactions on Automatic Control*, 2013.

[155] N. Chatzipanagiotis, Y. Liu, A. Petropulu, and M. M. Zavlanos, "Controlling groups of mobile beamformers," in *IEEE CDC*, 2012.

[156] Y. Huang, W. He, K. Nahrstedt, and W. C. Lee, "Requirements and system architecture design consideration for first responder systems," in *Technologies for Homeland Security, 2007 IEEE Conference on*, IEEE, 2007.

[157] P. N. Pathirana, N. Bulusu, A. V. Savkin, and S. Jha, "Node localization using mobile robots in delay-tolerant sensor networks," *Mobile Computing, IEEE Transactions on*, 2005.

[158] P. Corke, C. Detweiler, M. Dunbabin, M. Hamilton, D. Rus, and I. Vasilescu, "Experiments with underwater robot localization and tracking," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007.

[159] L. Hu and D. Evans, "Localization for mobile sensor networks," in *ACM MOBICOM*, 2004.

[160] "Are mobile tv and video still killer apps?." http://www.mobileindustryreview.com/2014/09/mobile-video-still-killer-app.html.

[161] "Mobile video is the new killer app for mobile operators." http://v-net.tv/2013/03/27/mobile-video-is-the-new-killer-app-for-mobile-operators/.

[162] "How self-driving cars could make streaming tv the next radio." https://goo.gl/Eg4RCu/.

[163] "Why iot, big data and smart farming is the future of agriculture." http://read.bi/2dSdFV2.

[164] "Wearable and big data: Potential challenges, potential rewards." https://goo.gl/UYiwLN/.

[165] "2020: Network for the networked society – an industry beyond smartphones." goo.gl/DdZk4n.

[166] "Mobile data traffic to increase 1000 times beyond 2020." http://channeleye.co.uk/mobile-data-traffic-to-increase-1000-times-beyond-2020/.

[167] "How to fix reception problems in cell phone dead zones." http://lsh.re/1BA7U.

[168] "Texas a & m fans set all-time record for data usage for at&t." goo.gl/7O8YRk.

[169] "Wireless insite." http://www.remcom.com/.

[170] "Hycopter drone uses hydrogen gas to fly for 4 hours." https://goo.gl/4Cayi8/.

[171] "Is a drone-balloon hybrid a match made in heaven?." http://nofilmschool.com/2016/07/why-use-panasonic-balloncam-over-pure-drone.

[172] "3d warehouse." https://3dwarehouse.sketchup.com/.

[173] P. R. Wolf and B. A. Dewitt, *Elements of Photogrammetry: with applications in GIS*, vol. 3. McGraw-Hill, 2000.

[174] Y. Dama *et al.*, "Mimo indoor propagation prediction using 3d shoot-and-bounce ray (sbr) tracing technique for 2.4 ghz and 5 ghz," in *EUCAP*, 2011.

[175] Z. Yun and M. F. Iskander, "Ray tracing for radio propagation modeling: principles and applications," *IEEE Access*, 2015.

[176] Y. Tao, H. Lin, and H. Bao, "Gpu-based shooting and bouncing ray method for fast rcs prediction," *IEEE Transactions on Antennas and Propagation*, 2010.

[177] "3d robotics - x8+ drones." https://3dr.com/wp-content/uploads/2016/02/X8-Operation-Manual-vC.pdf.

[178] "Securifi almond." https://www.securifi.com.

[179] "Top 1% of mobile users consume half of worlds bandwidth, and gap is growing." https://nyti.ms/2ioEoOp.

[180] F. Rashid-Farrokhi, K. R. Liu, and L. Tassiulas, "Transmit beamforming and power control for cellular wireless systems," *IEEE JSAC*, 1998.

[181] "Google loon." https://www.solveforx.com/loon/.

[182] "Facebook internet drones." https://www.wired.com/2016/07/facebooks-giant-internet-beaming-drone-finally-takes-flight/.

[183] "Darpa landroids." https://defense-update.com/products/l/landroids.htm.

[184] "Vehicle mounted antenna mast." https://goo.gl/wrx4c2.

[185] A. Dhekne, M. Gowda, *et al.*, "Cell tower extension through drones: Poster," in *Poster, Mobicom*, 2016.

[186] M. Gowda, N. Roy, and R. R. Choudhury, "Infrastructure mobility: A what-if analysis," in *ACM Hotnets*, 2014.

[187] M. Gowda, A. Dhekne, and R. Roy Choudhury, "The case for robotic wireless networks," in *WWW*, 2016.

[188] A. Y. Chung, J. Jung, *et al.*, "Poster: Swarming drones can connect you to the network," in *Mobisys*, 2015.

[189] G. Durgin *et al.*, "Measurements and models for radio path loss and penetration loss in and around homes and trees at 5.85 ghz," *IEEE Transactions on Communications*, 1998.

[190] L. J. Greenstein *et al.*, "A new path-gain/delay-spread propagation model for digital cellular channels," *IEEE Transactions on Vehicular Technology*, 1997.

[191] V. Erceg, L. J. Greenstein, S. Y. Tjandra, S. R. Parkoff, *et al.*, "An empirically based path loss model for wireless channels in suburban environments,"

[192] K. R. Schaubach, N. Davis, *et al.*, "A ray tracing method for predicting path loss and delay spread in microcellular environments," in *IEEE VTC*, 1992.

[193] T. Rautiainen *et al.*, "Verifying path loss and delay spread predictions of a 3d ray tracing propagation model in urban environment," in *IEEE VTC*, 2002.

[194] G. Liang and H. L. Bertoni, "A new approach to 3-d ray tracing for propagation prediction in cities," *IEEE Transactions on Antennas and Propagation*, 1998.

[195] K. Rizk *et al.*, "Two-dimensional ray-tracing modeling for propagation prediction in microcellular environments," *IEEE Trans on Vehicular Tech.*, 1997.

[196] M. Catedra *et al.*, "Efficient ray-tracing techniques for three-dimensional analyses of propagation in mobile communications: application to picocell and microcell scenarios," *IEEE Antennas and Propagation Magazine*.

[197] V. Namboodiri and L. Gao, "Prediction-based routing for vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, 2007.